ISSN 2007-9737

Benefits and Challenges of Military Artificial Intelligence in the Field of Defense

Jairo Eduardo Márquez-Díaz*

Universidad de Cundinamarca, Facultad de Ingeniería,
Colombia

jemarquez@ucundinamarca.edu.co

Abstract. The article explores the growing integration of AI in the military sphere. It highlights its potential to improve efficiency and accuracy in field operations, predictive analytics, and logistics due to its ability to process large volumes and learn autonomously. The study also emphasizes the role of AI in analyzing data from various sources such as satellite images, intelligence information and social media, which helps in target identification and operation of autonomous vehicles. It also addresses important ethical and legal challenges, including transparency in decision-making and liability for errors or collateral damage. It raises concerns about the vulnerability of military systems to attack, their tampering and the potential for accidental damage. The research employs a qualitative review of existing literature on types of AI and applications in the military context, focusing on benefits and risks. The study aims to synthesize key issues related to military AI capabilities, oversight, and governance, providing directions and precautions for responsible development of AI in defense.

Keywords. Artificial intelligence, algorithms, military, neural networks, technology.

1 Introduction

Artificial Intelligence (AI) is being increasingly used in the military field to improve efficiency and precision in field operations, predictive analysis, and logistics, among others. The ability to process large volumes of data in real time and to learn autonomously makes AI have great potential in defense and national security.

However, its use also raises significant ethical and legal challenges, such as transparency in decision-making and liability in case of errors or collateral damage.

Some civilian technology companies, particularly defense contractors, are actively involved in the field of military-grade AI. Though there are concerns that military AI systems could end up vulnerable to attack and tampering or cause accidental damage on a large scale. There are also questions about responsibility and transparency under the current geopolitical instability, especially when considering that the US, China, Russia, Iran, Turkey [4] and Israel, among other countries [32], lead the development of AI for large-scale military purposes.

The potential of military-grade AI is vast and diversified, the importance of which lies in being able to analyze large amounts of data from different sources, such as satellite imagery, intelligence signals, communications, and social media, to identify patterns and trends that may be relevant to national security. With this capability, target identification from surveillance images and video provides| critical information on the battlefield.

This brings us to the use of AI in weapon systems and autonomous vehicles such as drones, ground vehicles, and air defense systems, where human intervention is minimal.

Therefore, the present study aims to examine the benefits and challenges of using artificial intelligence in military operations and defense systems. To this end, the research question is formulated: What are the main advantages and ethical concerns associated with the integration of artificial intelligence capabilities in military domains? The methodology is framed in the qualitative review of the existing literature on applications of artificial intelligence in the military context.

The review will analyze the documented benefits of AI for defense activities, as well as the ethical, legal and security risks highlighted by experts and researchers. Key themes related to military AI capabilities, oversight, and governance will be synthesized to identify promising directions and precautions for responsible defense AI development.

2 Military Grade Artificial Intelligence

Military-grade artificial intelligence refers to AI systems developed and used specifically for military and defense applications [16]. These AIs are designed to meet the rigorous requirements and challenges associated with the military environment [5], where one seeks to gain a decisive advantage over potential adversaries. Some characteristics of this type of AI are detailed below:

– Robustness and Reliability: These AIs are designed to work reliably and withstand harsh conditions. They must be able to operate in a diverse range of environments and conditions often under extreme stress. This means that they must operate in hostile environments and resist physical damage, such as combat areas, and maintain their functionality even in the presence of interference or cyber-attacks [6].

– Processing capacity: These AIs are designed with advanced processing capabilities to ensure proper handling of large volumes of data. They can perform complex information analysis and processing tasks quickly and efficiently. This means that its design and development in many cases is tailored.

– Autonomy: Many military applications require AI systems that can operate autonomously, without the need for constant human supervision. This involves tasks like navigating a drone across a battlefield, identifying, and tracking targets, or managing logistics and supply chains.

– Adaptability: These types of AI are highly adaptable and can adjust to different situations and changing scenarios on the battlefield.

They can learn and update their models and algorithms as new data is obtained and circumstances change.

– Decision making: These AI systems often need to make decisions in real time, due to the changing dynamics of information. This requires advanced algorithms and high-performance computing power.

– Interoperability: These systems often need to integrate with others, just like military platforms. Consequently, it is required that they be designed with interoperability in mind, using standard protocols and interfaces whenever possible, since, due to their security characteristics, they must work under robust encrypted communication protocols such as AES-256, OMEMO or ZRTP.

– Security and confidentiality: Given the sensitive nature of their tasks, military-grade AI systems must have rigorous security and confidentiality standards. This includes encryption, secure communication channels, and measures to prevent tampering or unauthorized access. The design of the chips must withstand cyberattacks, data corruption, equipment failures and other threats, because they will be used in high-risk military operations.

– The design of the chips must withstand cyber-attacks, data corruption, equipment failure and other threats, because they will be used in high-risk military operations.

– Ethical considerations: The use of this type of AI raises a few ethical considerations. These systems must be designed in a way that they respect international law and minimize the risk of harm to civilians [29].

– Human-machine interaction: These AIs can operate autonomously or in collaboration with human operators. Human-machine interaction is essential in the military environment, where operators can harness the power of AI processing and analysis to support informed decision-making.

Table 1. Characteristics of a military-grade artificial intelligence

Feature	Description
Design	– Specifically designed for military applications. – Incorporates advanced algorithms and machine learning capabilities. – Adapts to challenging environments such as extreme weather conditions, interference, and cyber-attacks.
Operability	– It can work autonomously or in collaboration with human operators. – Integrated into military platforms and systems such as unmanned vehicles, surveillance systems and weapons. – Follow strict security and confidentiality protocols to protect sensitive information.
Functionality	– Recognition and processing of images and video in real time. – Analysis and evaluation of intelligence data for decision making. – Ability to predict and model scenarios. – Support in the identification and response to threats in real time.
Applications	– Surveillance and reconnaissance on the battlefield. – Perimeter defense. – Elimination of explosives with robots and drones. – Locating and tracking enemy targets. – Support in search and rescue operations. – Data analysis for the generation of military strategies. – Simulations and virtual training for military personnel. – Prediction and prevention of failures in military equipment and systems. – Smart cameras for facial identification. – Tactical loitering in urban warfare. – Synthetic aperture radar. – DNA profiling algorithms. – Intelligence recovery. – Asymmetric war. – Cybersecurity to protect digital assets and critical infrastructure. – Cyber defense and cyber-attacks. – Disinformation from deepfakes manipulating audios and images. – Prediction of weather patterns.

Table 1 summarizes the characteristics of a military-grade AI in terms of design, operability, functionality, and applications under a general vision. This is because its implementation and exact specifications vary according to the needs and requirements of each system, be it military or intelligence agency.

Other additional details about this type of AI are that they often have access to massive amounts of data for training purposes, including military intelligence, surveillance data, and information from conflicts and war games. This allows them to learn much faster and reach higher levels of performance.

For this to be possible, cutting-edge artificial intelligence techniques are required that go beyond what is commercially available. Some developed by companies and military contractors specialized in AI, whose specific regulations and policies are focused on applications related to defense and national security.

3 Technology and Artificial Intelligence

AI-related technology presents its own developments and challenges, such is the case of AI chips, specialized for different functions within the armed and security forces [34, 23]. For example, AI for air superiority and defense uses vision systems and machine learning to identify and track targets. In the case of vehicle autonomy, it focuses on navigation, perception, and decision-making, etc.

Other applications include:

- Surveillance and reconnaissance: They are used to analyze and process images and videos in real time with the aim of detecting and recognizing objects, people, or threats on the battlefield or in the civil sphere with national security interests.
- Decision making: AI can analyze large volumes of intelligence data, providing valuable information in making strategic and tactical decisions.
- Support military operations: AIs can assist in threat identification and response, as well as search and rescue operations.
- Simulation and training: AIs can provide realistic simulations and virtual training for military personnel, allowing them to practice scenarios and improve their skills without real risk. To do this, it uses technologies such as augmented, virtual, and mixed reality, telepresence, and virtual worlds or metaverses [18].

Here are some examples of AI weapons already in use:

- Harpy: Is an Israeli kamikaze drone that uses AI to autonomously seek out and destroy radar emitters and air defense systems.
- The SGR-A1 is a South Korean automated turret gun robotic system manufactured by Samsung. It uses AI to detect human targets and attack them with firearms without human intervention. It is deployed along the Korean Demilitarized Zone.
- Sky Warrior/Predator XP: It is an AI-powered drone from the US with improved automation and autonomy. This drone can launch attacks without human supervision. It is used for targeted attacks against high-value targets.
- Mantis: Is an AI-guided manned machine gun developed by the American company SparkCognition. It uses computer vision to automatically detect and track human targets and guide the gunner's aim without manual adjustments.
- A10-AJ: Is an AI upgrade developed by Boeing for existing A10 Warthog attack aircraft. It uses computer vision to identify targets faster than humans. Allows pilots to take on more threats in less time. It still requires a human operator to launch weapons.
- Informant V2: Is an AI target recognition system developed by BAE Systems. It is designed to help soldiers identify potential threats and focus their attention on what matters most in combat situations.
- LOCUST: Is a system made up of a swarm of AI-enabled drones being developed by the US Air Force. The swarm can autonomously detect and identify targets and then coordinate an attack with minimal human intervention [21]. There are similar initiatives being developed by other countries.
- XQ-58A Valkyrie: Tactical air-to-air and air-to-surface autonomous aircraft, which works with AI/ML for various operations. It is the first fully operational sixth generation aircraft.
- Missiles: Several countries are developing AI-guided missiles that can autonomously identify and track moving targets. Including the US Perdix microdrones swarm and China's DR-8 missiles.
- Tanks: The superpowers are investigating the use of AI to control tanks with greater autonomy, this includes automatic detection of

targets, navigation through various types of terrain and choosing optimal routes.

- Submarines: AI and automation are being explored for underwater operations to handle complex tasks such as sensor data analysis, combat systems management and threat assessment. This technology aims to reduce the size of the required crew and increase the degree of autonomy.

In fact, there is the first autonomous submarine, Boeing's XLUUV (Orca) that integrates these characteristics.

- Ships: Unmanned AI ships are being developed for missions such as reconnaissance, mine detection and attack. Examples of this are the US Navy's Sea Hunter and Sea Hawk, and Norway's Black Hornet drone.
- Artillery Systems: Some artillery guns are combined with AI and targeting automation, allowing them to spot, classify, and engage moving targets in seconds instead of minutes.
- Roadrunner: Is a reusable vertical take-off and landing (VTOL) autonomous aerial vehicle (AAV). Its function is air and ground defense, in which it can launch, identify, intercept, and destroy various types of air threats. It uses multiple AI systems that allow a single operator to monitor swarms of these AAVs, creating autonomous collaborative platforms in the process.
- Rifles: AI-assisted prototype sniper rifles can detect human targets, calculate range, wind, and suggest shot location.

A variant used by US police is ShotSpotter, which uses specialized sensors and AI algorithms to detect gunshots in real time. These sensors can pinpoint the exact location of the shot and alert agencies in less than a minute. Also, machine learning algorithms are used to confirm if the recorded sound is indeed a gunshot.

- Cyberglobes: Is a tool that detects criminal activities such as financial fraud, drug trafficking, cyber-attacks, and terrorist activities in real time through the analysis of data generated in social networks and conversations. In addition, the system has additional functions such as data mapping and geolocation [7].
- Sky shield: advanced Israeli multipurpose electronic warfare system incorporated into combat aircraft. Deploy countermeasures to counter threats, creating safe corridors for aircraft squadrons for defense and attack.
- Drone Dome: is an anti-aircraft system against drones that nullifies them using electronic jammers and advanced AI that allows you to quickly locate these targets and nullify them using a laser beam director.
- Imilite: Is a system designed for battlefield intelligence, surveillance and reconnaissance that integrates multiple sensors and platforms for intelligent and efficient exploitation. It specializes in the centralization and unification of processing and exploitation of various types of data such as images and videos and other types of intelligence.
- MXSERVER: Is a video and photography media analysis tool that uses facial recognition and machine learning, which allow to identify suspects and provide useful information to security agencies.
- Sea Breaker: Missile specialized in selective impacts using AI algorithms and artificial vision.
- Iron Vision: Is a high-definition artificial vision system incorporated into tanks, which allows operators to observe the surroundings with a 360-degree angle. This allows you to locate targets more precisely, with situational awareness powered by AI algorithms.
- POWER, Persistent Optical Wireless Energy Relay or Energy Transmission Initiative. Wireless power transmission to troops and equipment on the battlefield, whose nodes are autonomous drones.
- Directed energy weapons: It consists of accurately tracking air or ground targets and destroying them with laser beam shots mediated by advanced AI software. There are various prototypes under development and testing, for example the 8x8 VBCR system of the North American Stryker family.

Table 2. Artificial neural networks most used in the military field

Neural Network	Description	Applications in the military field
Convolutional Neural Networks (CNN)	Designed to process structured data, such as images or audio signals.	Detection and classification of objectives in images and videos, facial recognition for identification of people, analysis and processing of satellite images, drones, and intelligent surveillance systems.
Recurrent Neural Networks (RNN)	They are capable of processing sequential data and have feedback connections, which allows them to have memory.	Natural language processing, such as social media sentiment analysis to identify opinions and attitudes, real-time machine translation of intercepted communications, automatic text generation in reports and analytics, intelligence data processing for pattern and threat detection.
Generative Adversarial Neural Networks (GAN)	They combine a generator and a discriminator to create synthetic and realistic samples.	Generation of synthetic data for training detection and recognition algorithms, creation of synthetic images and videos for simulations and virtual training, generation of virtual adversaries to test and improve the security of defense and cybersecurity systems.
Transforming Neural Networks- Transformers	It employs attentional mechanisms used in natural language processing.	Recognition and translation of languages in intercepted communications, analysis, and classification of documents to identify relevant information, intelligence information processing to identify patterns and threats, automatic report generation to provide real-time intelligence.
Neural Networks for Sequence Analysis (Seq2Seq)	These networks are applied in the development of multilingual communication, particularly in the collection and exchange of information between different groups and nations.	Automatic translation of documents and communications in different languages with the aim of facilitating the exchange of information between different groups and nations in military settings or between intelligence agencies.
NeuRBF: A Neural Fields Representation with Adaptive Radial Basis Functions	It uses general radial bases in images with flexible position and shape, which allows the AI greater spatial adaptability and better adjustment to target signals, enlarging it without losing resolution.	It uses high-precision imaging systems and analysis of 2D images and 3D fields from various sources, with the ability to regenerate grids and/or render for analysis of specific objects.
Neural Networks for Anomaly Detection	Designed to detect unusual or anomalous patterns in large data sets.	Detection of suspicious activities or potential threats in surveillance data, network traffic or intrusion detection systems, critical infrastructure protection and defense systems.

- ChatGPT Military: OpenAI removed language prohibiting the use of its technology for military purposes from its usage policy, which previously included a ban on weapons development and warfare. While current OpenAI tools cannot be used directly for violence, they can assist in a variety of military-related tasks, such as document processing and analysis.
- Policy changes indicate a potential shift toward supporting operational infrastructure for military use (Biddle, 2024). Equivalently, China is employing a version like ChatGPT called Baidu's Ernie Bot and iFlyTek's Spark to train military AIs in automated combat simulations.

There is a wide range of autonomous weapons and systems whose functionality falls directly to AI,

which are currently in development or have limited operational use, apart from clandestine research by the military and private contractors that do not come to light publicly. While most still require human assistance for operation, the trend is clearly toward greater autonomy and less human involvement over time.

For example, fifth and sixth generation fighter aircraft have built-in AI for reconnaissance, detection, planning and conventional and electronic attack with three-dimensional vision, eliminating the pilot from making decisions when multiple variables are presented that can confuse him in the field of combat. battle, such as attacking multiple targets and/or evading air-to-air or ground-to-air missile attacks, or sharing data between different ships, even accompanying drones.

A particularity of this technology is its scalability, that is, it can be updated and new developments incorporated into existing ones without having to start new developments from scratch, such is the case of integrated military systems; addressing open sensor system architectures, open modular sets of C5ISR/EW(CMOSS) standards and airborne capability environments, among others, involving technologies such as advanced radar systems, avionics, navigation, electronic warfare, signals intelligence , communications and other mission-critical military systems.

4 AI Algorithms

In the military industry, there are various artificial intelligence algorithms that are widely used for various applications. Some of them are mentioned below:

1. Artificial Neural Networks (ANNs): These are a type of machine learning algorithm inspired by the human brain. ANNs can be used for a variety of tasks, such as image recognition, Natural Language Processing (NPL), and speech recognition, among others. They are characterized by being a type of computational model inspired by the structure and functioning of the human brain.

   These networks are composed of multiple interconnected artificial neurons. Each neuron receives inputs that are processed by a set of activation functions, then produces conditioned outputs that are reprocessed recursively in the hidden layers until an ideal output is obtained.

   During the training of a neural network, the weights of the connections between the neurons are adjusted so that the network can learn to correctly map the inputs to the desired outputs. Learning algorithms, such as error backpropagation, are used to iteratively adjust the weights to improve network performance. The following table shows some types of ANN most used in the military field.

2. Support Vector Machines (SVM*)*: Is a machine learning algorithm that can be used for classification and regression tasks. SVMs work by finding a hyperplane that separates the data into two classes by maximizing the margin between the data classes. The training data is mapped to a higher dimensional space by a kernel function, which allows nonlinear hyperplanes to be found.

   During training, the support vectors, which are the closest data points to the separation hyperplane, are selected. During the classification stage, the new data is mapped to the same high-dimensional space and assigned to a class based on its location.

3. Decision trees: They are supervised learning algorithms used for classification and regression tasks. Decision trees work by constructing a tree-like structure that represents the relationships between certain characteristics and the target variable. It is also related to random forest learning algorithms, which combine multiple decision trees to improve prediction accuracy.

4. Variational autoencoders (VAEs): These are a class of deep learning models that combine the power of neural networks with probabilistic graphical models. They consist of two main components: an encoder network and a decoder network, where VAEs learn from input data that can be sampled to generate new data that is like it. VAE can learn to generate new images or data that are like the training data.

5. Computer vision*:* It is a branch of AI focused on allowing machines to interpret and understand visual data from the world around them. It is used for tasks such as object detection and tracking, facial recognition, and autonomous navigation.
6. Natural Language Processing (NLP): It is a technology that allows you to summarize complex texts easily and quickly. It uses algorithms to identify and/or generate main ideas and summarize them in a shorter text, just like translators and chat bots. This technology can be used in various industries, ranging from education and journalism to electronic espionage systems.
7. Cognitive computing: Deals with the creation of intelligent computing systems with the ability to reason, learn, solve problems, and make decisions without human intervention. In the military field it is used in image processing, strategic planning, logistics and cybersecurity.
8. Multimodal models: Can understand and process multiple types of data simultaneously, such as text, images, audio, or multimodal combinations. This type of AI encompasses the previously mentioned models, plus others that are under development such as Large Language Models as Optimizers [19], which seeks to improve the performance of large language models (LLM) by optimization through PROmpting (OPRO); which is nothing more than meta-indications described in natural language that generates plausible solutions based on the description of a problem and the previous solutions.

5 Advanced AI

The field of artificial intelligence is constantly evolving, with new advances and technologies emerging all the time. These are some of the latest trends and projections in military AI extended to the civilian field with certain limitations [9]:

- Machine Learning (ML): It is a branch of artificial intelligence "that focuses on developing algorithms and models that allow computers to learn and improve automatically from data, without being explicitly programmed for each specific task" [1] Algorithms are trained to find patterns and correlations from large data sets that lead to decision making and forecasts.

- Deep Learning (DL): it specializes in training multilayer artificial neural networks so that they learn hierarchical representations of data. It has been a major driver of advanced AI research and applications in recent years. DL algorithms in conjunction with convolutional neural networks (CNN) and recurrent neural networks (RNN), have achieved remarkable results in various domains, including computer vision, NLP, and speech recognition.

  A variant of DL is analog deep learning based on neuromorphic computing, where programmable resistors are used that work in an equivalent way to transistors. These resistors form matrix layers that in turn create complex neural networks and analog synapses that work millions of times faster than biological synapses.

- Generative AI: Involves the use of AI models to generate new content, such as images, text, and audio. Recent advances in generative models, such as generative adversarial networks (GANs) and transformer models, have led to significant advances in realistic image synthesis, text generation, and creative applications. These models can create highly compelling and diverse results, pushing the boundaries of AI-generated content.

A recent development is Voicebox from the company Meta which is a generative AI that speaks. The system with a few seconds of audio (Flow Matching) can create a dialogue with a tone of voice identical to that of a human being.

Although it is a private and restricted system, it allows, in principle, apart from translating and suppressing background noise, impersonating the voice of personalities and, therefore, creating very well-crafted scams and deepfakes.

Developments like this are being incorporated into drones [12], where AI and voice recognition technology are integrated to make them self-sufficient.

- Large Language Models (LLM): Are a type of AI trained with a large amount of text-like data. They can generate human-like text based on the input they receive. In the military field, these models are used in a variety of ways:
  - Planning and Strategy: They assist in military planning by providing information, generating scenarios, and suggesting strategies based on historical data. They also help visualize and describe complex problems, making the planning process more efficient.
  - Data analytics: They allow you to analyze large amounts of text data, such as intelligence reports, and extract relevant information to identify patterns, trends, and threats that human analysts might miss.
  - Simulation and Training: Can be used to create realistic training scenarios for military personnel, generate dialogue, scenarios, and responses that help train soldiers for various situations.
  - Communication: They help in the translation of languages, being particularly useful in international military operations. They can also generate clear and concise reports, summaries, and other forms of communication.
  - Decision Making: By providing data-driven insights and predictions, LLMs can support decision-making processes in the military. They can help assess the potential results of different strategies and actions.
- Generative AI in the data cloud: Consists of advanced generative AI services, including chatbots and intelligent search systems. An example of this service is the Snowflake platform with NVIDIA, which fuses high-performance ML and AI with large volumes of proprietary, structured data. Using this data, custom generative AI models can be built, ranging from hundreds of terabytes to petabytes of raw information, to create and tune custom LLMs that power specific applications and services. These services encompass advertising, media and entertainment, financial services, health care and life sciences, manufacturing, retail and consumer packaged goods, technology, and telecommunications.
- Reinforcement Learning (RL): It is an area of AI that focuses on training agents to make sequential decisions through interactions with an environment. The RL has made significant advances in solving complex problems, including games, robotics control, and autonomous systems. Recent developments in RL algorithms, such as deep reinforcement learning and model-based RL, have shown impressive performance in challenging domains, leading to advances in autonomous vehicles, industrial automation, and more.
- AI and Big Data: AI uses algorithms and ML to analyze large amounts of data to generate information that enables strategic planning and tactical execution. Big Data refers to the massive volumes of data that are generated within military operations, including sensor data, satellite imagery, and intelligence reports.

  The fusion of these technologies improves decision-making processes and operational efficiencies, helping to identify patterns and trends in data that may not be immediately apparent to human analysts. For example, they can be used to analyze data from various sources in real time, such as sensors, Unmanned Aerial Vehicles (UAVs), and soldiers to better understand the battlefield, weather conditions, and enemy movements.
- AI and cybersecurity: Machine learning algorithms can detect patterns and anomalies that indicate potential threats, making them a valuable tool in the event of a cyberattack [20]. AI and ML are increasingly used in cybersecurity applications to detect threats, analyze risks, and protect networks. Some key areas where AI is used in cybersecurity include:
  - Malware detection: AI can identify malicious files and behavior that are unknown to or have eluded traditional antivirus software. This helps detect new and sophisticated cyberattacks. The military uses AI-enabled malware detection

to protect its networks and communication systems.

- Intrusion detection: AI systems can monitor network traffic for anomalies and suspicious patterns that indicate a cyber-attack or security breach is taking place.
- Threat intelligence: With high processing power, AI can analyze large volumes of data on cyber threats to identify patterns, associate threats with actors, and predict future attacks [11]. This threat intelligence helps the military strengthen its defenses against specific adversaries.
- Vulnerability management: AI systems can identify and prioritize vulnerabilities in software, which helps organizations and the military patch vulnerabilities before they can be exploited by attackers.
- Cybercrime: Some experts [36, 13] argue that AI will also enable more sophisticated cyberattacks and weapons in the future. The military has an interest in developing defensive and offensive AI-powered cyber capabilities.

- Explainable AI (XAI): Aims to improve the interpretability and transparency of AI systems. While DL's models have achieved remarkable performance, their decision-making processes are often viewed as black boxes. Recent developments in XAI research focus on the development of techniques and algorithms to provide explanations and insights into the predictions and decisions of AI models. This is crucial to build trust, address bias, and ensure ethical and responsible AI systems, whose actions can be easily understood by humans.
- AI and Edge Computing: Edge Computing involves processing data where it is generated (i.e., at the "edge or perimeter" of the network), rather than in a centralized location. By leveraging edge computing, AI models can be deployed directly to devices enabling privacy-sensitive and real-time AI applications. This includes applications in autonomous vehicles, Internet of Things (IoT) devices with all their variants [17] and smart city infrastructure, where low latency, privacy and bandwidth constraints are critical considerations.
- AI that designs chips: It is a scalable technology that aims to create processors automatically. As a particular case at the time of writing this article, there is Qimeng No 1 technology, which exceeds 4000 times the intelligence of ChatGPT4. Although this technology is far from designing chips at scales equivalent to high-performance chips, the proof of concept has been passed, so getting custom processors is only a matter of time.
- Neurotechnology: Research is being done on the ideal creation of a brain-computer interface, implants, and neural prostheses. Integrating neurotechnology with AI provides opportunities to collect vast amounts of neural data that can be used to train artificial intelligence systems [31].

  For example, data from brain-computer interfaces can be used to identify neural patterns that correspond to certain thoughts, intentions, or actions. This data helps improve algorithms that can interpret and react to neural signals. For example, Augmented Cognition: Employs neural interfaces that may one day be used to improve cognitive abilities such as focus, memory, and situational awareness for soldiers and intelligence personnel.
- Neurotechnology in the military and intelligence field has several potential applications such as lie detection. There is research on using brain data to detect deception more accurately than traditional polygraph tests.

  There is brain reading, where it is speculated that one day it will be possible to decode complex thoughts directly from brain activity, which could have important implications for security [22], surveillance and intelligence gathering.

  However, this application is still highly controversial and speculative now [24]. Brain-controlled drones and robots are another potential application, where experiments have

shown that subjects can control drones and robotic arms using only their brain signals.

This could allow a soldier to operate military equipment remotely without compromising his personal integrity on the battlefield. Work is also underway to improve prosthetics through neural interfaces that provide more natural control in injured soldiers.

- Organoid intelligence: Its goal is to create standardized brain organoids to develop biological artificial intelligence, since current AI systems have scaling problems that require increasing amounts of energy and data, while the human brain, on the other hand, uses progressive learning [33].

The bioengineering that supports this research aims to cultivate standardized human brain organoids with glial cells, using microphysiological systems to recreate brain architecture and functionality and take advantage of them with interface technologies and artificial intelligence [28]. In the end, what is expected is to provide information to the organoids and measure their output leading, for example, to training organoids for tasks like playing video games or controlling robots.

In terms of projections, AI is expected to continue to become more integrated into the military field. We will likely see more personalized experiences thanks to AI, as well as improvements in civilian areas like healthcare, transportation, and education. However, these advances will also bring challenges, particularly around issues like privacy and security.

6 Discussion

While the general characteristics of military AI are like commercial AI in terms of machine learning, computer vision, etc., the scope, performance, security, ethics, and role within the military make these systems distinct from what is commercially available. Added to the fact that the resources allocated to invest in this technology are greater and, therefore, expands the boundaries of AI technology as stated.

There are ethical questions surrounding the use of AI for lethal purposes. Some argue that AI should only be used for defensive or support functions, and not to carry out attacks. However, others believe [25, 14] that AI weapons will be inevitable, and the focus should be on ensuring that they are used responsibly [22].

Some experts [35, 26] argue that humans should always maintain ultimate control over any weapons system involving AI. Others believe that full autonomy is inevitable and necessary to keep up with technological advances from potential threats [3, 15], especially when geopolitical dynamics today are increasingly unstable.

Another aspect to consider is the potential use of military AI to spy on civilians. Although espionage is not new, the use of AI for this purpose is new, with the particularity that not only these actions are carried out by the militia, but also by civilians. This is an example that military AI developments are inevitably making their way into civilian life in various contexts that can negatively affect privacy, surveillance, and other aspects of daily life. Recently, a study by [10] has exposed how military AI for use in intelligence operations to identify terrorist cells is being used for monitoring and surveillance of employees, especially in the United States.

The objective is to carry out a data analysis to identify organizers where the possibility of labor strikes arises and act against them. For example, locating organizers so that their employers can fire them before they form a union. This system can be used by employers during the personnel recruitment process, to avoid hiring future union organizers or people who have had a problem in places where they previously worked.

The problem does not end here, since some AIs are used for emotion detection, which still has flaws, proven biases, discrimination, and wrong assumptions. This means that people can be falsely accused. This scenario currently lacks regulation and, therefore, several companies are using this type of technology without any control or guarantee of transparency. It follows that as AI becomes more integrated into our lives, questions related to ethics and standards become more important.

This includes issues such as data privacy, the possibility of algorithmic bias, and the impact of AI

on jobs; demanding continuous research and development in security and exploration of methods to ensure the safe and reliable implementation of AI systems. Looking to the future, military AI has several highly relevant developments and projections, with the potential to improve exponentially, following the pattern of Moore's Performance Law [27].

This implies high computational power and data available to train AI models that doubles approximately every two years. As it is, there are continuous advances in deep learning, with the exploration of more sophisticated architectures, optimization techniques, and training strategies to tackle even more complex tasks.

This advancement goes hand in hand with the continuous improvement of unsupervised learning and self-supervised learning [30, 2], making it easier for military AI systems to learn from unlabeled data and reduce their reliance. to large datasets. These developments have also brought an increased focus on ethical and responsible AI practices [8]. This includes addressing issues like bias, fairness, transparency, and the robustness of AI systems. At this stage, the integration of AI with other emerging technologies is being explored.

This includes neurotechnology, neuromorphic computing, and quantum computing. These integrations envision new possibilities for improving the computational power and algorithmic capabilities of AI. For example, they have the potential to revolutionize diagnosis, treatment, and healthcare delivery by ensuring interoperability and security in critical systems, improving healthcare overall, discovering new drugs, and scaling personalized medicine to new levels. While the technologies have great potential to transform many fields, there are also ethical concerns regarding their use in corporate, societal, and global military and surveillance.

AI when making autonomous decisions can have consequences in people's lives. This entails establishing an ethical and legal framework for its use in the military field and clearly defining responsibility in case of errors or collateral damage. Future developments will need to be balanced with consideration of privacy, security, and human rights. Thus, the reliance on AI in military operations can be a risk if systems fail or are compromised by adversaries.

At this point, AI can be susceptible to bias and discrimination, triggering negative consequences in military operations by undermining confidence in critical systems. It is also important that the military proactively address these challenges and work on solutions that enable the responsible and effective use of AI.

7 Conclusions

Military-grade AIs are designed to meet the needs of the military in various areas, considering advanced processing capabilities, adaptability, and security, among other aspects. Its application ranges from recognition and surveillance to predictive analysis for strategic decision making. Its objective is to improve the effectiveness and performance of military operations such as autonomous vehicles (such as drones or unmanned submarines), cyber defense, logistics and supply chain management, among others.

AI is increasingly playing a crucial role in improving cybersecurity defenses for the military. By automating tedious tasks, accelerating threat detection, and assisting with threat intelligence, AI technologies can help the military more effectively protect their digital assets and critical infrastructure against cyberattacks.

While fully autonomous weapons may not yet exist, there are many examples of AI enhancing existing weapons systems by providing detection, prioritization, tracking, and target recommendation capabilities, but the ultimate decision to engage and use lethal force still it falls to a human operator for now. There is an ongoing military artificial intelligence arms race among various nations, which raises numerous pressing ethical and security considerations that must be addressed as this technology continues to develop.

Notably, much of this technological advancement is occurring outside of public scrutiny. It should also be acknowledged that increasing geopolitical tensions between major world powers are accelerating efforts to develop advanced AI for both cybersecurity defense and potentially offensive cyber capabilities, further exacerbating an arms race in this domain.

Consequently, as AI systems are increasingly deployed for purposes of both cyber defense and

cybercrime, careful international governance and oversight will be vital in the future to help manage the complex ethical and security implications of progress in these technologies.

Acknowledgments

The author thanks the University of Cundinamarca for its support and funding. Faculty of Engineering, Chia, Colombia.

References

1. **Álvarez, E. (2023).** Machine learning: Así funciona la disciplina que les enseña a las computadoras a aprender por sí mismas. https://tn.com.ar/tecno/novedades/2023/05/29/machine-learning-asi-funciona-la-disciplinaque-les-ensena-a-las-computadoras-a-aprender-por-si-mismas/E.
2. **Balestriero, R., Ibrahim, M., Sobal, V., Morcos, A., Shekhar, S., Goldstein, T., Goldblum, M. (2023).** A cookbook of self-supervised learning. arXiv preprint arXiv:2304.12210. DOI: 10.48550/arXiv.2304.12210.
3. **Blanchard, A., Floridi, L., Taddeo, M. (2022).** The doctrine of double effect & lethal autonomous weapon systems. Available at SSRN 4308862. DOI: 10.2139/ssrn.4308862.
4. **Can, M. (2023).** Under the leadership of our president:'Potemkin AI'and the Turkish approach to artificial intelligence. Third World Quarterly, Vol. 44, No. 2, pp. 356–376. DOI: 10.1080/01436597.2022.2147059.
5. **Evron, Y., Bitzinger, R. A. (2023).** The fourth industrial revolution and military-civil fusion: A new paradigm for military innovation? Cambridge University Press.
6. **Gaba, S., Budhiraja, I., Kumar, V., Martha, S., Khurmi, J., Singh, A., Singh, K., Askar, S., Abouhawwash, M. (2024).** A systematic analysis of enhancing cyber security using deep learning for cyber physical systems. IEEE, Vol. 12, pp. 6017–6035. DOI: 10.1109/ACCESS.2023.3349022.
7. Gandharv**, K. (2021).** Best reliable deep-tech to track criminals. https://analyticsindiamag.com/best-reliable-deep-tech-to-track-criminals/.
8. **Gill, A. (2019).** Artificial intelligence and international security: the long view. Ethics & International Affairs, Vol. 33, No. 2, pp. 169–179. DOI: 10.1017/S0892679419000145.
9. **Gregory, T. H. X., Chuan, N. S., Bingquan, S. (2022).** Self-supervised learning with deep neural networks for computer vision. In: Guo, H., Ren, H., Wang, V., Chekole, E. G., Lakshmanan, U. (eds) IRC-SET 2021, Springer, Singapore. DOI: 10.1007/978-981-16-9869-9_47.
10. **Grill, G., Sandvig, C. (2023).** Military AI's next frontier: your work computer. https://www.wired.com/story/military-ais-next-frontier-your-work-computer/
11. **Johnson, J. (2019).** Artificial intelligence & future warfare: implications for international security. Defense & Security Analysis, Vol. 35, No. 2, pp. 147–169. DOI: 10.1080/14751798.2019.1600800.
12. **Johnson, J. (2020).** Artificial intelligence, drone swarming and escalation risks in future warfare. The RUSI Journal, Vol. 165, No. 2, pp. 26–36. DOI: 10.1080/03071847.2020.1752026.
13. **Johnson, J. (2020).** Artificial intelligence: A threat to strategic stability. Strategic studies quarterly, Vol. 14, No. 1, pp. 16–39.
14. **Koch, W. (2022).** AI for aerospace and electronic systems: Technical dimensions of responsible design. IEEE Aerospace and Electronic Systems Magazine, Vol. 38, No. 1, pp. 106–111. DOI:10.1109/MAES 2022.3228300.
15. **Lamnabhi-Lagarrigue, F., Samad, T. (2023).** Social, organizational, and individual impacts of automation. In: Nof, S.Y. (eds) Springer Handbook of Automation, pp. 61–75. DOI: 10.1007/978-3-030-96729-1_3.
16. **Laudien, T., Ernst, J., Schmerwitz, S. (2023).** Bringing a colored head-down display symbology heads up: display fidelity review of a low-cost see-through HMD. Artificial Intelligence and Machine Learning for Multi-

Domain Operations Applications, Vol. 12538, pp. 191–197. DOI: 10.1117/12.2664840.

17. **Márquez, J. (2021).** Internet de las cosas (IoT) y grandes datos frente ataques de denegación de servicio distribuido (DDoS). UMNG, VI Congreso Internacional de Administración de la Seguridad y Salud en el Trabajo. Gestión del riesgo: una visión global e integral. Editorial Neogranadina. pp. 189–235. DOI: 10.18359/litgris.6278.
18. **Márquez, J. (2020).** Virtual world as a complement to hybrid and mobile Learning. International Journal of Emerging Technologies in Learning, Vol. 15, No. 22, pp. 267–274. DOI: 10.3991/ijet.v15i22.14393.
19. **Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q., Zhou, D., Chen, X. (2023).** Large language models as optimizers. Google DeepMind, pp. 1–42. DOI: 10.48550/arXiv.2309.03409.
20. **Márquez, J. (2021).** Dronica as an option for the security and defense of cities. Academia Letters, No. 861, pp. 1–3. DOI: 10.20935/AL861.
21. **Márquez, J. (2023).** Desarrollos tecnológicos e implicaciones de los drones autónomos militares: perspectivas en la geopolítica mundial. Revista Tecnológica, Vol. 35, No. 1, pp. 137–151. DOI: 10.37815/rte.v35n1.1018.
22. **Maschmeyer, L. (2022).** Subverting skynet: the strategic promise of lethal autonomous weapons and the perils of exploitation. 14th International Conference on Cyber Conflict: Keep Moving! (CyCon), Vol. 700, pp. 155–171. DOI: 10.23919/CyCon55549.2022.9811008.
23. **Montasari, R. (2023).** National artificial intelligence strategies: A Comparison of the UK, EU and US approaches with those adopted by state adversaries. Countering Cyberterrorism: The Confluence of Artificial Intelligence, Cyber Forensics and Digital Policing in US and UK National Cybersecurity, Vol. 101, Cham: Springer International Publishing, pp. 139–164. DOI: 10.1007/978-3-031-21920-7_7.
24. **Müller, O., Rotter, S. (2017).** Neurotechnology: Current developments and ethical issues. Frontiers in Systems Neuroscience, Vol. 11, No. 93, pp. 1–5. DOI: 10.3389/fnsys.2017.00093.
25. **Nadibaidze, A. (2022).** Great power identity in Russia's position on autonomous weapons systems. Contemporary security policy, Vol. 43, No. 3, pp. 407–435. DOI: 10.1080/13523260.2022.2075665.
26. **Pekarev, J. (2023).** Attitudes of military personnel towards unmanned ground vehicles (UGV): a study of in-depth interview. Discover Artificial Intelligence, Vol. 3, No. 1, pp. 24. DOI: 10.1007/s44163-023-00058-4.
27. **Qian, S., Liu, M., Dou, Y., Fink, Y. Yan, W. (2023).** Una ley de Moore para fibras permite tejidos inteligentes. Revista Nacional de Ciencias, Vol. 10, No. 1, DOI:10.1093/nsr/nwac202.
28. **Quirión, R. (2023).** Brain organoids: are they for real? Frontiers in Science, Vol. 1, p. 1148127. DOI: 10.3389/fsci.2023.1148127.
29. **Raaijmakers, S. (2019).** Artificial intelligence for law enforcement: Challenges and Opportunities. IEEE Security & Privacy, Vol. 17, No. 5, pp. 74–77.
30. **Rani, V., Nabi, S. T., Kumar, M., Mittal, A., Kumar, K. (2023).** Self-supervised learning: A succinct review. Archives of Computational Methods in Engineering, Vol. 30, No. 4, pp. 2761–2775. DOI: 10.1007/s11831-023-09884- 2.
31. **Sattler, S., Pietralla, D. (2022).** Public attitudes towards neurotechnology: Findings from two experiments concerning brain stimulation devices (BSDs) and brain-computer interfaces (BCIs). PloS One, Vol. 17, No. 11, p. e0275454. DOI: 10.1371/journal.pone.0275454.
32. **Scollick, A. (2023).** The Irish defence forces in the drone age. The EU, Irish Defence Forces and Contemporary Security, Cham: Springer International Publishing, pp. 295–314. DOI: 10.1007/978-3-031-07812-5_15.
33. **Smirnova, L., Morales, P. I., Hartung, T. (2023).** Organoid intelligence (OI) the ultimate functionality of a brain microphysiological system. ALTEX Alternatives to animal experimentation, Vol. 40, No. 2, pp. 191–203. DOI: 10.14573/altex.2303261.

34. **Speith, T., Speith, J., Becker, S., Zou, Y., Biega, A., Paar, C. (2023).** Expanding explainability: From explainable artificial intelligence to explainable hardware. DOI: 10.48550/arXiv.2302.14661.
35. **Taddeo, M., Blanchard, A. (2022).** Accepting moral responsibility for the actions of autonomous weapons systems: A moral gambit. Philosophy & Technology, Vol. 35, No. 3, p. 78. DOI: 10.1007/s13347-022-00571.
36. **Yamin, M., Ullah, M., Ullah, H., Katt, B. (2021).** Weaponized AI for cyberattacks. Journal of Information Security and Applications, Vol. 57, p. 102722. DOI: 10.1016/j.jisa.2020.102722.

Article received on 16/08/2023; accepted on 19/02/2024.
**Corresponding author is Jairo Eduardo Márquez-Díaz.*

ISSN 2007-9737

Detection of Wildlife Species in the Peruvian Amazon Using Transfer Learning

Luis Alberto Holgado-Apaza[1,*], Ruth Nataly Aragon-Navarrete[2], Coren Luhana Ancco-Calloapaza[3], Edgar E. Carpio-Vargas[4], Marleny Quispe-Layme[5], José Miguel Barrón-Adame[6], Rafael Guzman-Cabrera[7], Wilian Quispe-Layme[8]

[1] Universidad Nacional Amazónica de Madre de Dios, Departamento Académico de Ingeniería de Sistemas e Informática, Peru

[2] Universidad Nacional Amazónica de Madre de Dios, Departamento Académico de Ecoturismo, Peru

[3] Universidad Nacional San Agustín de Arequipa, Peru

[4] Universidad Nacional del Altiplano, Departamento Académico de Estadística e Informática, Peru

[5] Universidad Nacional Amazónica de Madre de Dios, Departamento Académico de Contabilidad y Administración, Peru

[6] Universidad Tecnológica del Suroeste de Guanajuato, Cuerpo Académico de Desarrollo Tecnológico Multidisciplinario, Mexico

[7] Universidad de Guanajuato, División de Ingenierías del Campus Irapuato-Salamanca, Mexico

[8] Universidad Nacional Amazónica de Madre de Dios, Departamento Académico de Educación y Humanidades, Peru

{lholgado, raragon, maquispe, wquispe}@unamad.edu.pe, canccoca@unsa.edu.pe, ecarpio@unap.edu.pe, guzmanc@ugto.mx, mbarrona@utsoe.edu.mx

Abstract. Wildlife holds an important role within the Amazon biome. However, wildlife identification and documentation methods in the Amazonian wilderness pose considerable challenges for fauna biology and ecology professionals. This complexity arises from the demand for specialized expertise and the substantial investment of time required. This challenge is compounded by the remarkable resemblance between various animal species. In this study, we delve into the feasibility of diverse iterations of the YOLO (You Only Look Once) algorithm in order to detect wildlife species in the Peruvian Amazon. Our assessment covers a spectrum of YOLO versions, including YOLOv5x6, YOLOv5l6, YOLOv7-W6, YOLOv7-E6, YOLOv8l, and YOLOv8x. To empower our models, we embarked on a training journey using a dataset comprising 653 images thoughtfully collected from reputable sources in ecology and tourism marketing. This dataset encompasses six species: Ara ararauna, Ara chloropterus, Ara macao, Opisthocomus hoazin, Pteronura brasiliensis, and

Saimiri sciureus. Our efforts show the efficiency of the YOLOv5l6 model, which stands out prominently in all metrics evaluated. This model achieves a Precision rate of 86.1%; Recall of 84.7%, F1-Score measuring 85.39%, and mean Average Precision (mAP) of 88.1%. Noteworthy is the fact that this model also boasts the swiftest training time among its counterparts, with a total 30.71 minutes. These findings offer promising prospects for refining our understanding of Amazonian wildlife species and establishing proactive measures to safeguard those that face potential vulnerability or endangerment. The YOLO algorithm's capabilities underscore the confluence of technology and ecological conservation, providing optimism for the preservation of the Amazon's intricate biodiversity.

Keywords. Wildlife species, Peruvian Amazon, YOLO, object detection, transfer learning.

1 Introduction

Amazon biome is one of the main sources of biodiversity in the world's ecosystems [1]. Wildlife is an important component within its territory [2, 3]. While the species list is growing all the time, only a fraction of the Amazon's enormous biodiversity is known to science [4, 5].

According to estimates only 90-95 per cent of mammals, birds and plants are known, only 2-10 per cent of insects have been described, and only 2,500 from the approximately 6,000 – 8,000 amazon fish species have been described [6]. Eight countries share responsibility for the Amazon, one of them is Peru which is home of 11.27 per cent of the biome. Peru stands out as one of the most biodiverse countries in the world.

The extensive Amazonian forests cover 62% of the Peruvian territory and they are home to approximately 50% of the plant species registered by the Ministry of Environment-MINAM [7]. This remarkable diversity also includes numerous wildlife endemic species in the region.

For example, 115 endemic bird species have been identified (representing 6% of the world total), 109 mammal species (27.5% of the world total), 185 amphibian species (48.5% of the world total) and 59 endemic butterfly species (12.5% of the world population) [8].

The efforts to conservate and manage fauna in the Amazon Biome does not still fill gaps of knowledge about tropical fauna [3, 9]. It is needed to strength accurate identification and monitoring of wildlife through discovering and documentation. Tradition-ally the identification is based on different biological assessments of biodiversity.

However, identification and documentation methods of wildlife in the Amazon represents a considerable challenge for professionals engaged in population biology and ecology studies, as it involves a high cognitive load and significant time consumption [10]. This difficulty is attributed to the existence of multiple types of animals that exhibit high similarity to each other, making difficult their precise classification [11].

Over the past few decades, automated species identification has brought about a revolution in conventional methodologies [12]. Recent research has demonstrated the emergent use of artificial intelligence (AI) and more specifically computer vision in the identification and monitoring biodiversity species, is the case of MobileNetV3 a deep learning model that were successfully used enabling faster and more efficient analysis at identifying mangrove species [13].

The same technology has been used to detect plant species [14]. Similar technologies have contributed to detect camels on roads [15], as well as to identify rodent species [16]. Another example is the use of AlexNet model to identify ringed seal of the Saimaa, according to results the experiment get an accuracy of 91.2% in the individual identification of species [17].

In the same line, [18] proposed a framework for animal's recognition which consisting of 2 Convolutional neural network (CNN) based models for image classification, results show values close to 90% in the identification of 3 most common animals. [19] proposed the application of a 3-branch VGG CNN in parallel with the aim of recognizing wild animal species. CNN has contributed to recognize wild boars [20] toads/frogs, lizards and snakes [21].

The transfer learning architecture called YOLOv3 has enabled the wildlife monitoring [11]. The advantages offered by CNN's applications in image recognition have been applied for different purposes [22, 23, 24], constructed the wildlife dataset of Northeast China Tiger and Leopard National Park to identify and process images captured with camera traps. Using three deep learning object detection models YOLOv5, FCOS

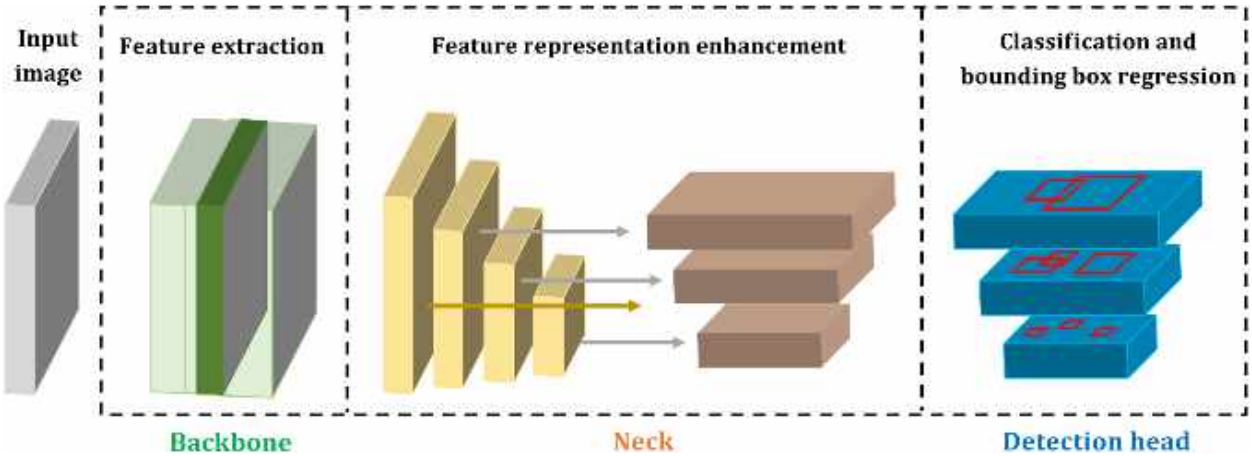


Fig. 1. General architecture of YOLO object detectors

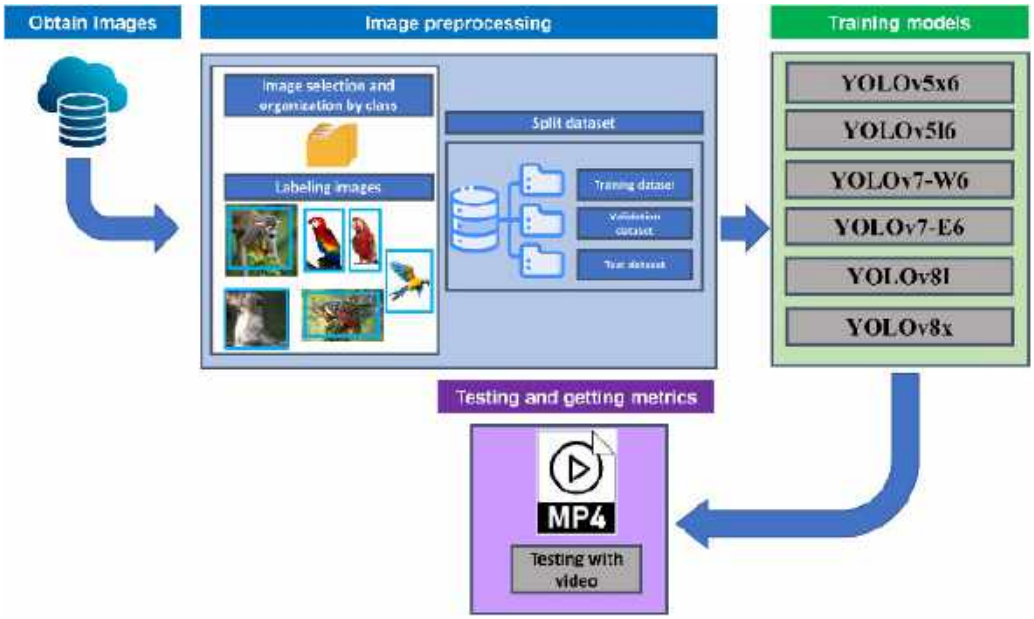


Fig. 2. Methodology used for the detection of wildlife species in the Peruvian Amazon

and Cascade R-CNN, they found an average Accuracy of 97.9%, along with an approximate mAP50 of 81.2% for all three models. Along the same lines, [25] used a transfer learning approach to detect the presence of four endangered mammals in the forests of Negros Island (Viverra tangalunga, Prionailurus javanensis sumatranus, Rusa alfredi and Sus cebifron).

The authors used the YOLOv5 model as a detection method. The trained model yielded a mean mAP50 of 91%. [26] proposed the identification of snake, lizard and toad/frog species from camera trap images using CNN. The results obtained by accuracy were 60% in the validation stage. [27] proposed a system to detect animals on the road and avoid accidents.

To do this, the animals were classified into groups of capybaras and donkeys. The authors used two variants of pre-trained CNN models: Yolov4 and Yolov4-tiny.

The results showed an accuracy of 84.87% and 79.87% for Yolov4 and Yolov4-tiny, respectively.

Table 1. Number of images per wildlife species

Species	Quantity
Ara ararauna	232
Ara chloropterus	50
Ara Macao	52
Opisthocomus hoazin	100
Pteronura brasiliensis	110
Saimiri sciureus	109
Total	653

Fig. 3. Image labeling process in the labeling tool

As shown, there are important advances at animal detection using deep learning technologies.

However, they have not been applied to detect wildlife species in the Peruvian part of the Amazon biome yet. The main objective of this study was to evaluate the application of the YOLO algorithm in its versions YOLOv5x6, YOLOv5l6, YOLOv7-W6, YOLOv7-E6, YOLOv8l and YOLOv8x in the detection of wildlife species in the Peruvian Amazon.

To achieve this, we have evaluated the aforementioned models using the following metrics: Precision, Recall, F1-Score and mAP50, applied to six species: Ara ararauna, Ara chloropterus, Ara macao, Opisthocomus hoazin, Pteronura brasiliensis and Saimiri sciureus.

In addition, as part of our contribution to the scientific community, we provide a labelled dataset for classification and/or detection of these species. We have structured the remaining contents of the paper as follows: In Section 2, the methodology adopted to conduct the experiments is presented in detail.

Results and discussions are addressed in Section 3, while our conclusions are presented in Section 4.

2 Material and Methods

We performed our experiments using the machine learning technique called transfer learning, which consists of using previously learned knowledge trained on large volumes of public images [28, 29]. Specifically, we have used the object detection algorithm in images and video called YOLO (You Only Look Once) in its versions YOLO-v5 [30], YOLO-v7 [31] and YOLO-v8 [32].

We trained and evaluated our models on a computer with these characteristics: AMD A12-9700P RADEON R7, 10 COMPUTE CORES 4C+6G at 2.50 GHz, 12 GB RAM, Windows 10 Home 64-bit operating system and x64 processor.

The development environment used was Google Colab with GPU accelerator type A100. Figure 1 shows the general architecture of YOLO, taking as reference the study presented by [33, 34]. The main components of YOLO are listed below:

- **Backbone**: The backbone is usually a convolutional neural network that extracts useful features from the input image [33].
- **Neck**: The neural network neck is used to extract features from images at different stages of the backbone. YOLOv4 make use of Spatial Pyramid Pooling (SPP) [35] and Path Aggregation Network (PAN) [36].
- **Head**: Is the final component of the object detector; this component is responsible for making the predictions from the features provided by the spine and neck [33].

In Figure 2, we show the methodology used for the detection of wildlife species in the Peruvian Amazon using transfer learning. It is composed of four phases, they are: Obtain images, images preprocessing, training models and testing and getting metrics. Each step of the proposed methodology is detailed below:

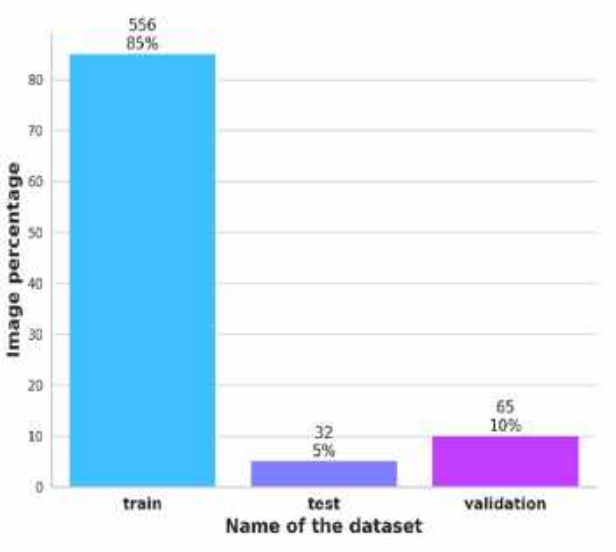


Fig. 4. Dataset division for learning transfer process

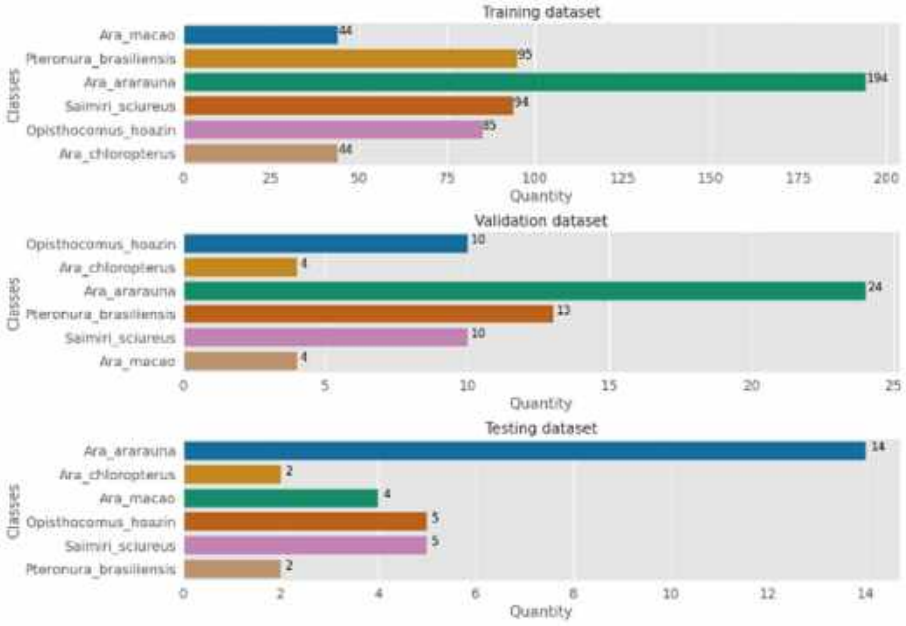


Fig. 5. Images distribution by classes in datasets training, validation and test

2.1 Image Obtaining

At this stage, we have searched for wild animals' images by their scientific name. The images were collected from websites related to ecology studies and tourism marketing; such as Rainforest expeditions [37], Go2peru [38], Ararauna Tambopata [39].

Then, in order to download the images with high resolution, we have used the Fatkun Batch Download Image extension in its version 5.7.7 by the Google Chrome [40], as it was used in previous

Table 2. Confusion matrix for three classes

	Prediction			
True	A	B	C	FN
A	n_{11}	n_{12}	n_{13}	$n_{12} + n_{13}$
B	n_{21}	n_{22}	n_{23}	$n_{21} + n_{23}$
C	n_{31}	n_{32}	n_{33}	$n_{31} + n_{32}$
FP	$n_{21} + n_{31}$	$n_{12} + n_{32}$	$n_{13} + n_{23}$	

Table 3. Configuration and training times for models

Model	Epochs	Batch	Input (resolution)	Training time in minutes
YOLOv5x6	70	16	640x640	57.94
YOLOv5l6	70	16	640x640	30.71
YOLOv7-W6	70	16	640x640	57.48
YOLOv7-E6	70	16	640x640	60.36
YOLOv8l	70	16	640x640	37.14
YOLOv8x	70	16	640x640	62.20

studies [41]. Finally, we have selected and filter manually only the images in jpg format.

2.2 Image Processing

This stage encompassed the process of image curation, organization and labeling. The curation of images was made according to the species to be identified, since the search yielded images related to the keyword. In Table 1, It is shown the total summary of images by species in the first dataset after selection.

We have performed the images labeling manually. For this purpose, we have used the labelImg tool [42]. In Figure 3, we show an example of this labeling task in the specie Saimiri sciureus. As a result of this process, a textual file is generated and it fulfills the mission to designate each image.

Internally this file encompasses both the class to which the image belongs and the coordinates delimiting the bounding boxes containing the image. Finally, we have divided the dataset as follow:

We divided the dataset into 85% (556 images) for training, 10% (65 images) for validation and 5% (32 images) for testing. Figure 4 shows this division graphically.

In Figure 5 we show a summary of the distribution of classes in the datasets used to train, validate and test the species detection models based on the YOLO architecture.

2.3 Trining Model

During this phase we conducted out our experiments with the object detection algorithm YOLO in its versions: YOLOv5x6, YOLOv5l6, YOLOv7-W6, YOLOv7-E6, YOLOv8l and YOLOv8x. Within the file called 'custom_data.yaml' we have defined the configuration of the path to the training, validation and test images.

In the same file, we have additionally configurated the classes as follow: Scientific names: [Ara_ararauna, Ara_chloropterus, Ara_macao, Opisthocomus_hoazin, Pteronura_brasiliensis, Saimiri_sciureus].

2.4 Testing and Getting Metrics

We performed our tests with a video adapted from the public videos: "Vive como sueñas | Reserva Nacional Tambopata" from the Ministry of Environment [43, 44] and the video "Manu & Tambopata" from the Antara-Peru travel agency [45]. Figure 6 and Figure 7, respectively, show a screenshot of these videos.

Fig. 6. Screenshot of video: "Vive como sueñas Reserva Nacional Tambopata"

Fig. 7. Screenshot of video: "Manu and Tambopata"

We have used the metrics: Precision, Recall, F1-Score and Mean Average Precision (mAP), and additionally the confusion matrix in order to evaluate the performance of the models. Subsequently we proceed to detail each of these metrics: Confusion matrix: It is an nxn, dimension table where n represents the number of classes or objects to be detected.

This metric allows to evaluate the performance of a classification algorithm by counting the hits and mistakes in each one of the model classes. A confusion matrix for three classes is observed in Table 2, which can be extrapolated to object detection and classification problems with n classes.

where:

FN represents false negatives.

FP represents false positives.

n_{11} represents the true positives (TP) for class A.

n_{22} represents the true positives (TP) for class B.

n_{33} represents the true positives (TP) for class C.

The true negatives (TN) for class:
A=$n_{22}+n_{33}+n_{32}+n_{23}$.
The true negatives (TN) for class:
B=$n_{11}+n_{33}+n_{31}+n_{13}$.
The true negatives (TN) for class:
C=$n_{11}+n_{22}+n_{21}+n_{12}$.

Precision: This metric, also known as positive predictive value (PPV) indicates the proportion of cases correctly identified as belonging to a specific class (e.g., class C) among all cases where the classifier claims to belong to that class.

In other words, accuracy answers the question: Considering that the classifier predicts that a sample belongs to class C, what is the probability that the sample actually belongs to class C? [46, 47]. Equation 1 illustrates the calculation of this metric:

$$\text{Precision}=\frac{\text{TP}}{\text{TP+FP}}, \tag{1}$$

where:

TP=True Positive.

FP=False Positive.

Recall: This metric, is also referred to as Sensitivity or True Positive Rate (TPR) measures the ratio of positive correctly identified positive cases (for our case study it represents the species to be identified) by the algorithm [48] Equation 2 shows the formula for calculating this metric:

$$\text{Recall}=\frac{\text{TP}}{\text{TP+FN}}. \tag{2}$$

F1-Score: It is defined as a harmonic mean of precision and recall. The F1 score reaches its best value at 1 and its worst value at 0. Equation 3 shows the formula for calculating this metric:

$$\text{F1}=2*\frac{Precision*Recall}{Precision+Recall}. \tag{3}$$

Average Precision (AP): This metric represents the relationship between precision and recall at different confidence thresholds, in addition to quantifying the ability of the detection model to discriminate between positive and negative classes. It is calculated from the Precision-Recall curve (PR Curve). Its value varies between 0 and 1, where an AP of 1 indicates perfect detection and AP of 0 indicates random detection [49, 51].

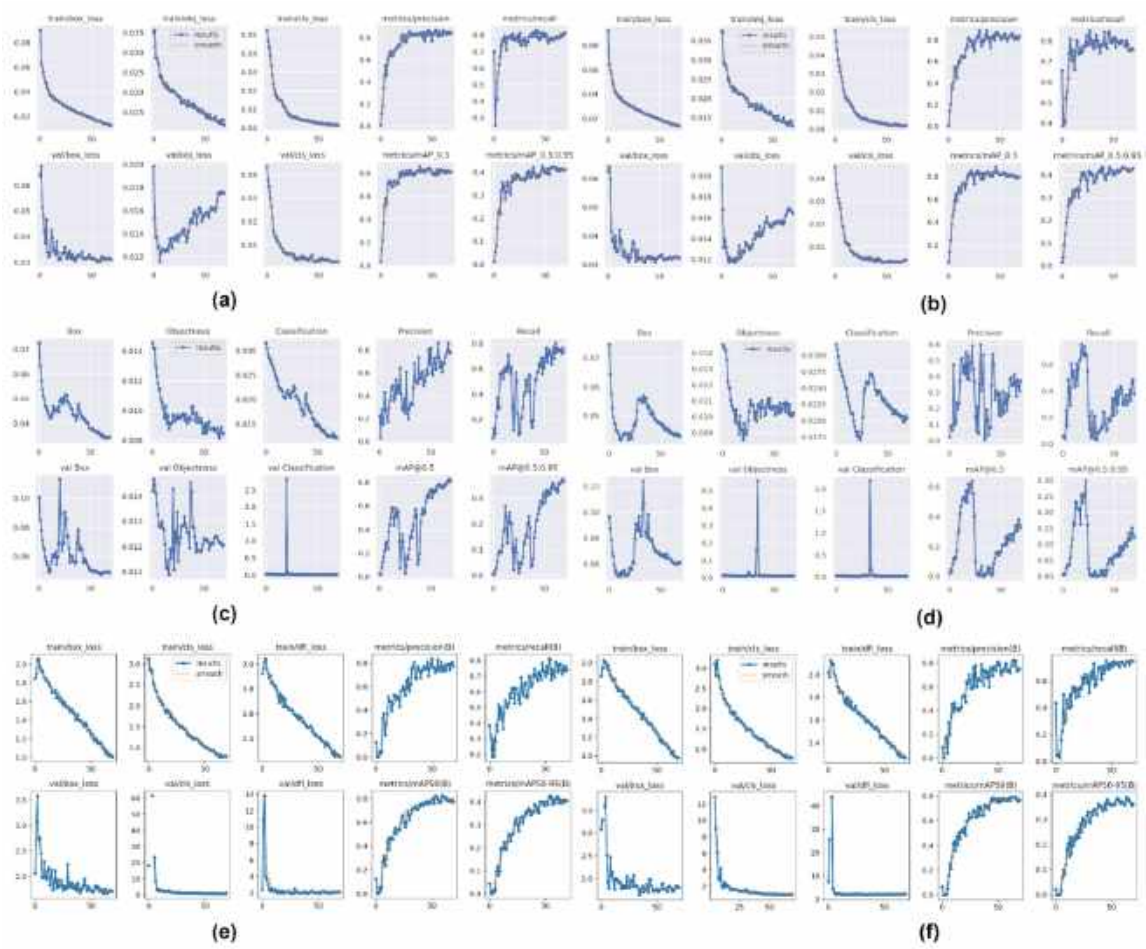


Fig. 8. Evaluation and performing metrics of the models during training. (a) YOLOv5x6 model, (b) YOLOv5l6 model, (c) YOLOv7-W6 model, (d) YOLOv7-E6 model, (e) YOLOv8l model and (f) YOLOv8x model

The mathematical operation for this calculation is shown in Equation 4:

$$AP_i = \int_0^1 P_i(R_i) dR_i, \tag{4}$$

where P and R refer to the precision and recall of the detection model that we detail in Equations (1) and (2), respectively. Mean average precision (mAP): We have calculated this metric by averaging the Average Precision (AP) values for all classes present in the dataset.

Its value fluctuates between o and 1, where 1 indicates perfect performance, i.e., all detections are correct and there are no false positives or false negatives. In Equation 5 we show the formula for the calculation of this metric:

$$mAP = \frac{1}{n} \sum_{i=1}^{n} AP_i. \tag{5}$$

3 Results and Discussions

Table 3, shows the configurations we have made for each YOLO version we used within our experimental framework, along with their corresponding training duration measured in minutes. Our selection process adhered to the guidelines outlined in the official documentation.

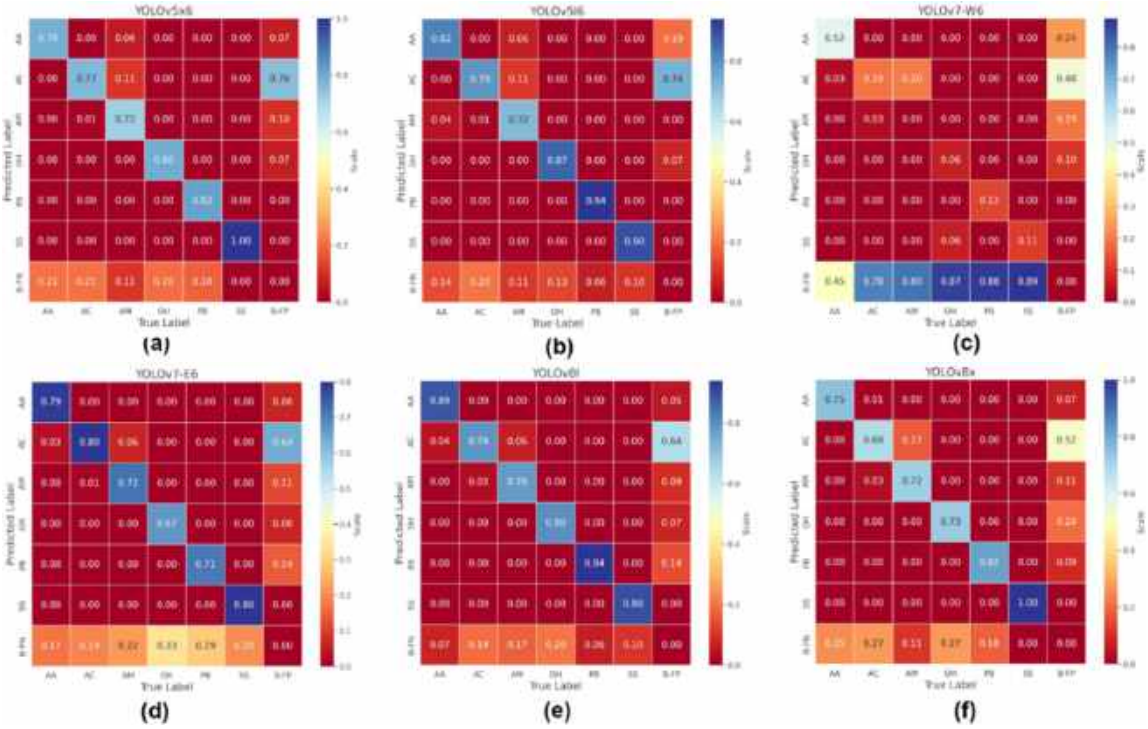


Fig. 9. Standard confusion matrices. (a) YOLOv5x6 model, (b) YOLOv5l6 model, (c) YOLOv7-W6 model, (d) YOLOv7-E6 model, (e) YOLOv8l model and (f) YOLOv8x model

According to this, we have chosen the two best pre-trained models per version [52]– [55]. We have made a particular exception for YOLOv7 because the YOLOv7-D6 and YOLOv7-E6E versions, which have a slightly better AP value, required a high computational cost for training, so we opted to use the YOLOv7-W6 and YOLOv7-E6 versions.

Figure 8 presents the metrics for evaluating and monitoring of the selected model's performance during training. The metrics are focused on the prediction accuracy of the object bounding boxes (box_loss) coordinates, error in the prediction of the classes of the detected objects (cls_loss), precision and recall.

It is noted that the YOLOv5 (Figure 8a and 8b) and YOLOv8 (Figure 8e and 8f) models exhibit remarkable stability and out-standing performance in the task of accurate bounding box localization. These models demonstrate a consistent tendency to reduce the loss associated with accuracy, which holds important relevance in image object detection applications.

Specifically, in our domain study related to wildlife species identification, the results obtained by YOLOv5 and YOLOv8 show a superior ability to accurately localize the bounding boxes of the interest objects.

Moreover, during the validation stage, it is observed that these models maintain their stability, which confirms their robustness and their potential for practical applications in the field of computer vision.

With regard to the "cls_loss" metric, which reflects the discrepancy between model's classification predictions and the actual labels of the object classes, it is graphically observed in Figure 8a and 8b that the YOLOv5 model manages to efficiently reduce this value down to epoch 50.

This suggests that as the model is trained its performance is better in the classification accuracy of wildlife species in the Peruvian Amazon.

Figure 9 shows the normalized confusion matrices during the training phase for the models utilized in our experiments. AA corresponds Ara

Table 4. Results obtained per evaluation metric

Model	Class	Precision	Recall	F1-Score	mAP50
YOLOv5x6	All	0.822	0.787	0.804	0.839
	Ara_ararauna	0.842	0.786	0.813	0.857
	Ara_chloropterus	**0.743**	0.760	**0.751**	0.790
	Ara_macao	**0.848**	**0.778**	**0.811**	**0.852**
	Opisthocomus_hoazin	0.819	0.733	0.774	0.747
	Pteronura_brasiliensis	0.812	0.763	0.787	0.812
	Saimiri_sciureus	0.870	0.900	0.885	0.978
YOLOv5l6	All	0.861	0.847	0.854	0.881
	Ara_ararauna	0.842	0.857	0.849	0.867
	Ara_chloropterus	0.727	0.745	0.736	**0.816**
	Ara_macao	0.781	0.722	0.750	0.759
	Opisthocomus_hoazin	**0.864**	**0.849**	**0.856**	**0.871**
	Pteronura_brasiliensis	**0.951**	**0.941**	**0.946**	**0.980**
	Saimiri_sciureus	**1.000**	**0.967**	**0.983**	**0.995**
YOLOv7-W6	All	0.371	0.378	0.374	0.327
	Ara_ararauna	0.326	0.571	0.415	0.514
	Ara_chloropterus	0.373	0.480	0.420	0.349
	Ara_macao	0.080	0.218	0.117	0.065
	Opisthocomus_hoazin	0.240	0.267	0.253	0.225
	Pteronura_brasiliensis	0.493	0.229	0.313	0.327
	Saimiri_sciureus	0.714	0.500	0.588	0.483
YOLOv7-E6	All	0.776	0.739	0.757	0.808
	Ara_ararauna	**0.851**	0.821	0.836	0.864
	Ara_chloropterus	0.700	**0.779**	0.737	0.765
	Ara_macao	0.753	0.722	0.737	0.792
	Opisthocomus_hoazin	0.832	0.662	0.737	0.791
	Pteronura_brasiliensis	0.613	0.647	0.630	0.681
	Saimiri_sciureus	0.909	0.800	0.851	0.952
YOLOv8l	All	0.743	0.806	0.773	0.817
	Ara_ararauna	0.828	**0.893**	**0.859**	**0.910**
	Ara_chloropterus	0.604	0.720	0.657	0.645
	Ara_macao	0.736	0.722	0.729	0.732
	Opisthocomus_hoazin	0.737	0.733	0.735	0.823
	Pteronura_brasiliensis	0.654	0.891	0.754	0.863
	Saimiri_sciureus	0.897	0.877	0.887	0.930
YOLOv8x	All	0.819	0.745	0.780	0.790
	Ara_ararauna	0.843	0.714	0.773	0.836
	Ara_chloropterus	0.711	0.624	0.665	0.698
	Ara_macao	0.820	0.758	0.788	0.795
	Opisthocomus_hoazin	0.784	0.726	0.754	0.694
	Pteronura_brasiliensis	0.856	0.765	0.808	0.820
	Saimiri_sciureus	0.898	0.885	0.891	0.898

ararauna species; AC, Ara cholopterus; AM, Ara macau; OH, Opisthocomus hoazin; PB, Pteronura brasiliensis and SS; Saimiri sciureus.

In this graphic it is also observed that YOLOv8l y YOLOv5l6 models attain the highest values along the main diagonal. This diagonal represents instances in which the predicted labels by the aforementioned models align with the actual labels and suggests that in our dataset of Peruvian Amazon wildlife, the YOLOv8l and YOLOv5l6 models effectively identify the six species studied. Table 4 presents the results of the obtained metrics

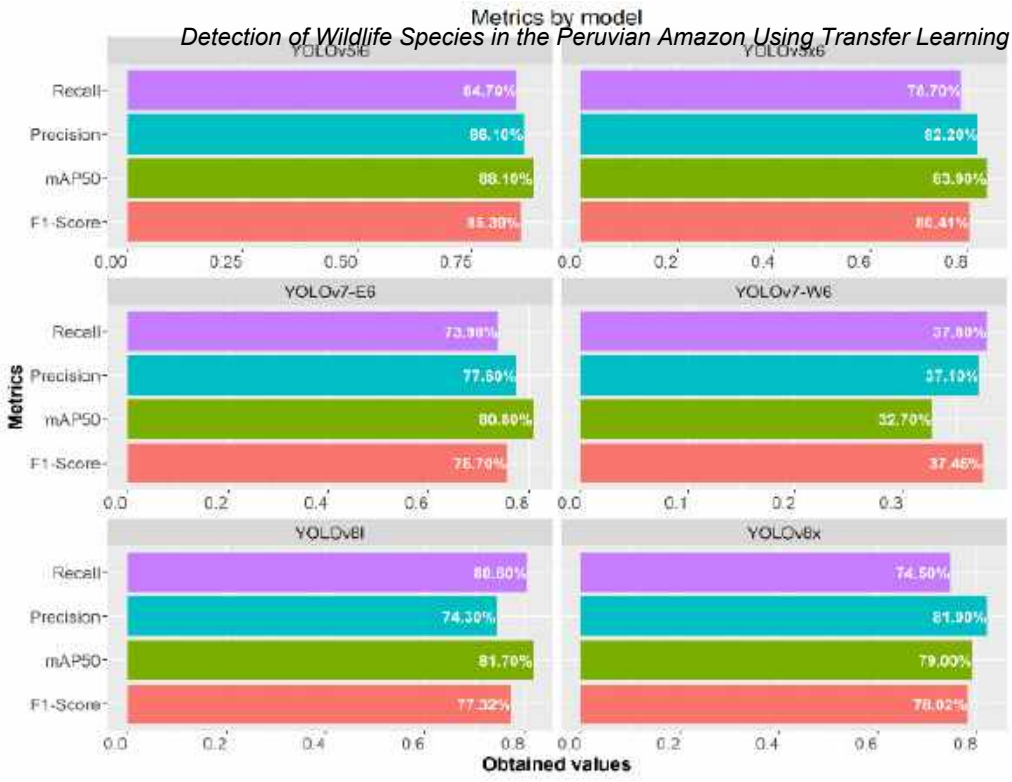


Fig. 10. Overview of the metrics for the models analyzed

for each model in our experiments in detection of the six Peruvian Amazon wildlife species. The best values per species and metric are highlighted.

It is important to stand out that the YOLOv5I6 model has demonstrated an outstanding performance by obtaining the highest values of Presicion, Recall, F1-Score and mAP50 for three of the six species analyzed: Opistho-comus hoazin, Pteronura brasiliensis and Saimiri sciureus.

The reference model demonstrated remarkable performance across evaluated metrics. It achieved a Precision rate of 86.4%, Recall of 84.9%, and an F1-Score of 85.6%, and a mAP50 of 87.1% for the first specie.

For the second one, the model exhibited exceptional Precision rate of 95.1%, paired with a Recall of 94.1%, an F1-Score of 94.6%, and an mAP50 of 98.0%.

The third specie yielded unprecedented results, attaining a perfect Precision of 100%, while sustaining a Recall of 96.7%, contributing to an impressive F1-Score of 98.3%. Remarkably, the mAP50 reached an astounding 99.5%. It's worth noting that this model consistently outperforms others, particularly evident in its remarkable mAP50 of 81.6% for the Ara chloropterus species.

Summarizing, our experiments have evidenced that the YOLOv5I6 model proves highly effective in specific species detection and it is notably excelling in the cases of Opisthocomus hoazin, Pteronura brasiliensis and Saimiri sciureus, such us in Ara chloropterus specie registering the highest mAP50 value.

On the other hand, the YOLOv5x6 model has also yielded remarkable results, particularly for the Ara macao species, in addition to obtaining higher values in the precision and F1 score metrics for Ara chloropterus. In Figure 10, we present a visual summary of the evaluated metrics for the six analyzed models. Our experiments highlight that the YOLOv5I6 model achieves the highest values in all of the evaluated metrics with a Precision rate of 86.1%, a recall of 84.7%, an F1-Score of 85.39% and an mAP50 of 88.1%.

Attained values are closely trailed by those reported by the YOLOv5x6 model. Notably, the overall averages of the metrics used in this study

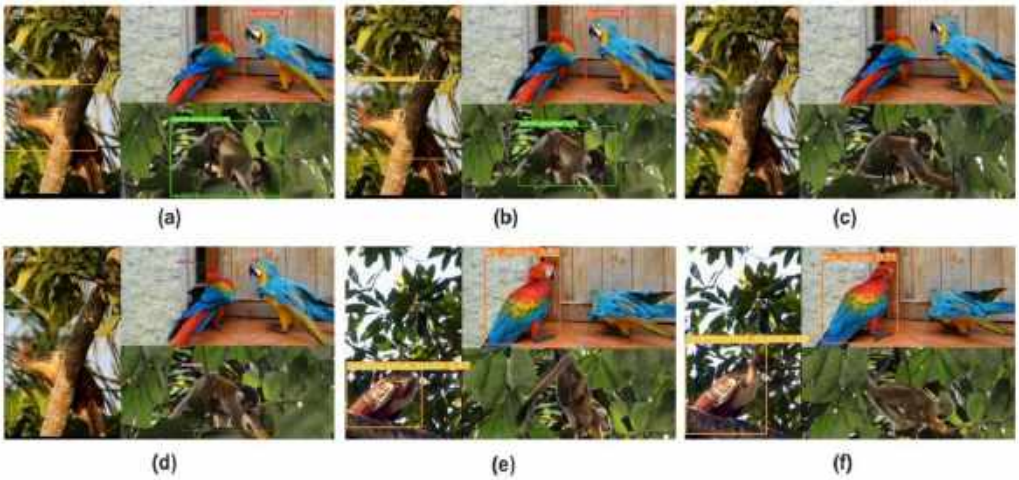


Fig. 11. Detection outcomes with confidence values within bounding boxes for models. (a) YOLOv5x6, (b) YOLOv5l6, (c) YOLOv7-W6, (d) YOLOv7-E6, (e) YOLOv8l and (f) YOLOv8x

firmly indicate the YOLOv5l model as the optimal choice for our specific case.

These findings resonate with the conclusions drawn by [24], who explored animal detection and classification through camera trap images. Their evaluation of YOLOv5, FCOS, and Cascade_R-CNN_HRNet32 models yielded an impressive average Precision of 97.9%, along with an approximate mAP50 of 81.2% across all models.

Likewise, [25] introduced a framework for detecting four endangered mammal species Viverra tangalunga, Prionailurus javanensis sumatranus, Rusa alfredi, and Sus cebifrons in the forests of Negros Island using the YOLOv5 model.

Their effort resulted in an average mAP50 of 91% and a commendable Precision of 91%. We posit that the disparities in the metric values observed are attributed to the size discrepancy between the species examined in our research and those in the referenced studies.

Specifically, the species within our study context are comparably smaller than those encompassed in the aforementioned research. Our findings hold high significance for applications in wildlife monitoring and conservation.

They underscore the effectiveness of the YOLOv5l6 model in detecting wildlife species within the expanse of the Peruvian Amazon. Furthermore, these outcomes establish robust basis for forthcoming research endeavors and initiatives aimed at safeguarding the biodiversity of Amazonian regions.

Figure 11 provides an enlightening overview of numerical results of confidence in bounding boxes in the detection of wildlife species in the Peruvian Amazon. These results are screenshot captures obtained from the execution of our models on a video file in mp4 format (minute 1:00), derived from the videos "Vive como sueñas | Reserva Nacional Tambopata" [56] and "Manu & Tambopata" [57].

It is noteworthy that the YOLOv7-E6 model is notable for its ability to identify the highest number of species in the video under analysis. However, a more detailed evaluation of the confidence associated with the object bounding boxes yields more accurate data.

We have observed that the YOLOv5l6 (b) model achieves a confidence of 83% for the Opisthocomus hoazin species, which represents the highest figure within this specific category. Concerning the detection of Ara arauna and Saimiri sciureus species, confidence levels of 90% and 75% are attained respectively.

It is worth noting that these values are remarkably close to those obtained by the YOLOv5x6 version. At the captured moment depicted in the figure, it is evident that both the YOLOv5l6 and YOLOv5x6 models failed to identify the Ara macao species.

However, it is important to note that the YOLOv7-E6, YOLOv8l and YOLOv8x models successfully detected this species at the same minute of capture. These findings provide a visual and tangible perspective on how the models perform in wildlife detection in the Peruvian Amazon.

These observations are crucial for understanding the applicability of the models in real world situations and highlight the significance of prediction confidence, as well as the selection of appropriate models based on detection objectives.

4 Conclusions

In this study we comprehensively assess the effectiveness of the YOLO algorithm in its versions YOLOv5x6, YOLOv5l6, YOLOv7-W6, YOLOv7-E6, YOLOv8l, and YOLOv8x.

Our evaluation centers on their suitability for detecting six wildlife species within the Peruvian Amazon, namely: Ara ararauna, Ara chloropterus, Ara macao, Opisthocomus hoazin, Pteronura brasiliensis, and Saimiri sciureus.

To build a robust foundation for our analysis, we curated a specialized dataset by sourcing images from ecological and tourism outlets, including Rainforest Expeditions, Go2Peru, and Ararauna Tambopata. The meticulous curation process, executed under the guidance of a wildlife specialist.

Our findings prominently showcase the prowess of the YOLOv5l6 model, which exhibits exceptional performance across all evaluated metrics. Notably, it achieves an impressive accuracy of 86.1%, a recall rate of 84.7%, an F1-Score of 85.39%, and a mean Average Precision (mAP) of 88.1%.

Remarkably, this model also boasts the shortest training duration at a mere 30.71 minutes among all models scrutinized. Furthermore, our experimental outcomes reveal a striking similarity between the achievements of the YOLOv5l6 model and those of the YOLOv5x6 model.

This convergence of results underscores the consistency and reliability of our evaluation methodology. These outcomes stand as promising and auspicious strides forward in fortifying initiatives aimed at identifying Amazonian wildlife species and vigilantly monitoring those that could potentially slip into states of vulnerability or endangerment.

By showcasing the potential of advanced algorithms, we not only demonstrate the power of technology but also emphasize the significance of collective efforts in safeguarding the rich biodiversity of the Amazon rainforest.

Data Availability Statement: We make our dataset and source code available to the scientific community at the following e-mail address[1] Additionally, you can watch the video used for the tests. There are also 6 videos that were generated for each model in the validation stage, these videos show the bounding boxes and their respective confidence value[2].

References

1. **Andrade-Silva, J., Baccaro, F. B., Prado, L. P., Guénard, B., Warren, D. L., Kass, J. M., Economo, E. P., Silva, R. R. (2022).** A large-scale assessment of ant diversity across the Brazilian Amazon Basin: Integrating geographic, ecological and morphological drivers of sampling bias. Ecography, Vol. 2022, No. 9, pp. e06295. DOI:10.1111/ecog.06295.
2. **Peres, E. A., Pinto-da-Rocha, R., Lohmann, L. G., Michelangeli, F. A., Miyaki, C. Y., Carnaval, A. C. (2020).** Patterns of species and lineage diversity in the Atlantic Rainforest of Brazil. Neotropical diversification: patterns and processes, pp. 415–447. DOI: 10.1007/978-3-030-31167-4_16.
3. **Geldmann, J., Manica, A., Burgess, N. D., Coad, L., Balmford, A. (2019).** A global-level assessment of the effectiveness of protected areas at resisting anthropogenic pressures. Proceedings of the National Academy of Sciences, Vol. 116, No. 46, pp. 23209–23215. DOI: 10.1073/pnas.1908221116.

[1] github.com/cheshire21/wild_species_detection_cnn

[2] www.youtube.com/@luisalbertoholgadoapaza919/videos

4. **de-Souza, V. D. C., de-Oliveira, R. E., Sais, A. C. (2022).** Agro e biodiversidade na agricultura familiar: potencial de diversificação e conservação em paisagens desmatadas na Amazônia. Desenvolvimento e Meio Ambiente, Vol. 60. DOI: 10.5380/dma.v60i0. 73625.
5. **Fidelis, E. G., Querino, R. B., Adaime, R. (2023).** The Amazon and its biodiversity: a source of unexplored potential natural enemies for biological control (Predators and parasitoids). Neotropical Entomology, Vol. 52, No. 2, pp. 152–171. DOI: 10.1007/s13744-022-01024-y.
6. **Charity, S., Dudley, N., Oliveira, D., Stolton, S. (2016).** Living Amazon, Report 2016. A regional approach to conservation', Brazilia and Quito, 2016.
7. **Torres-Montenegro, L. A., Ríos-Paredes, M. A., Pitman, N. C., Vriesendorp, C. F., Hensold, N., Mesones-Acuy, Í., Trujillo-Calderón, W. (2019).** Sesenta y cuatro nuevos registros para la flora del Perú a través de inventarios biológicos rápidos en la Amazonía peruana. Revista Peruana de Biología, Vol. 26, No. 3, pp. 379–392, DOI: 10.15381/RPB. V26I3.16780.
8. **SERNANP (2016).** 'PERÚ: PAÍS MEGADIVERSO', 2022. https://old.sernanp. gob.pe/sernanp/archivos/imagenes/vida/Peru-Pais Megadiverso.pdf.
9. **Dudley, N., Phillips, A., Amend, T., Brown, J., Stolton, S. (2016).** Evidence for biodiversity conservation in protected landscapes. Land, Vol. 5, No. 4, p. 38. DOI: 10.3390/land5040038.
10. **Pollicelli, M. D., Delrieux, C. A., Coscarella, M. A. (2020).** Avances en foto-identificación automatizada de fauna silvestre. Repositorio Institucional CONICET Digital, No. 5, DOI: 10.33414/AJEA.5.719.2020.
11. **Manasa, K., Paschyanti, D. V., Vanama, G., Vikas, S. S., Kommineni, M., Roshini, A. (2021).** Wildlife surveillance using deep learning with YOLOv3 model. 2021 6th International Conference on Communication and Electronics Systems ICCES, pp. 1798–1804. DOI: 10.1109/ICCES51350.2021. 9489121.
12. **de-Souza, V. L., Costa, F. B., Martins, T. F., de-Oliveira, P. R., Lima, J., Guimarães, D. P., Alencar-dos-Santos, E., Oliveira-de-Moura, M. N., Pereira-Sato, T., Pais-Borsoi, A. B., Bitencourth, K., Ribamar-Lima-de-Souza, J., Salles-Gazeta, G., Guilherme, E., Glauco-de-Araújo-Santos, F. (2023).** Detection of Rickettsia tamurae-like and other spotted fever group rickettsiae in ticks (Acari: Ixodidae) associated with wild birds in the Western Amazon, Brazil. Ticks and Tick-borne Diseases, Vol. 14, No. 4, p. 102182. DOI: 10.1016/j.ttbdis.2023.102182.
13. **Viodor, A. C. C., Aliac, C. J. G., Santos-Feliscuzo, L. T. (2023).** Identifying mangrove species using deep learning model and recording for diversity analysis: A mobile approach. 2023 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), pp. 1–6. DOI: 10.1109/eStream 59056.2023.10134992.
14. **Carranza-Rojas, J., Mata-Montero, E. (2017).** Identificación automática de especies de plantas de Costa Rica usando visión por computadora. Memorias de congresos TEC. III Jornadas Costarricenses de Investigación en Computación e Informática. DOI: 10.18845/ MCT.V0I0.4527.
15. **Alnujaidi, K., AlHabib, G. (2023).** Computer vision for a camel-vehicle collision mitigation system. Vol. 12, No. 1, pp. 141–149. DOI: 10. 5121/ijci.2023.120111.
16. **Seijas, C., Montilla, G., Frassato, L. (2019).** Identificación de especies de roedores usando aprendizaje profundo. Computación y Sistemas, Vol. 23, No. 1, pp. 257–266. DOI: 10.13053/CYS-23-1-2906.
17. **Nepovinnykh, E., Eerola, T., Kälviäinen, H., Radchenko, G. (2018).** Identification of Saimaa ringed seal individuals using transfer learning. Advanced Concepts for Intelligent Vision Systems: 19th International Conference, LNCS, Vol. 11182, pp. 211–222. DOI: 10.1007/978-3-030-01449-0_18.
18. **Nguyen, H., Maclagan, S. J., Nguyen, T. D., Nguyen, T., Flemons, P., Andrews, K., Ritchie, E. G., Phung, D. (2017).** Animal recognition and identification with deep convolutional neural networks for automated

wildlife monitoring. 2017 IEEE international conference on data science and advanced Analytics, Vol. 2018, pp. 40–49. DOI: 10.1109/DSAA.2017.31.

19. **Favorskaya, M., Pakhirka, A. (2019).** Animal species recognition in the wildlife based on muzzle and shape features using joint CNN. Procedia Computer Science, Vol. 159, pp. 933–942, DOI: 10.1016/J.PROCS.2019.09.260.
20. **Silva, L. C., Pádua, M. B., Ogusuku, L. M., Keese-Albertini, M., Pimentel, R., Backes, A. R. (2021).** Wild boar recognition using convolutional neural networks. Concurrency and Computation: Practice and Experience, Vol. 33, No. 22. DOI: 10.1002/CPE.6010.
21. **Binta-Islam, S., Valles, D., Hibbitts, T. J., Ryberg, W. A., Walkup, D. K., Forstner, M. R. (2023).** Animal species recognition with deep convolutional neural networks from ecological camera trap images. Animals, Vol. 13, No. 9, p. 1526. DOI: 10.3390/ANI13091526.
22. **Gomez Villa, A., Salazar, A., Vargas, F. (2017).** Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. Ecological Informatics, Vol. 41, pp. 24–32. DOI: 10.1016/j.ecoinf. 2017.07.004.
23. **Tan, M., Chao, W., Cheng, J. K., Zhou, M., Ma, Y., Jiang, X., Ge, J., Yu, L., Feng, L. (2022).** Animal detection and classification from camera trap images using different mainstream object detection architectures. Animals, Vol. 12, No. 15, p. 1976. DOI: 10.3390/ani12151976.
24. **Castañeda, J. A. J., De-Castro, A. L., Sy, M. A. G., AlDahoul, N., Tan, M. J. T., Karim, H. A. (2022).** Development of a detection system for endangered mammals in Negros Island, Philippines using YOLOv5n. International Conference on Computational Science and Technology Singapore: Springer Nature Singapore, Vol. 983, pp. 435–447. DOI: 10.31219/osf.io/62vje.
25. **Islam, S. B., Valles, D. (2020).** Identification of wild species in Texas from camera-trap images using deep neural network for conservation monitoring. 2020 10th Annual Computing and Communication Workshop and Conference IEEE, pp. 537–542. DOI: 10.1109/CCWC47524.2020.9031190.
26. **Sato, D., Zanella, A. J., Costa, E. J. X. (2021).** Computational classification of animals for a highway detection system. Brazilian Journal of Veterinary Research and Animal Science, Vol. 58, pp. e174951–e174951. DOI: 10.11606/ISSN.16784456.BJVRAS.2021.174951.
27. **Alnujaidi, K., AlHabib, G. (2023).** Computer vision for a camel-vehicle collision mitigation system. Vol. 12, No. 1, pp. 141–149. DOI: 10.5121/ijci.2023.120111.
28. **Abel, L. A. J., Oconer, T. C. N., Cruz, J. C. D. (2022).** Realtime object detection of pantry objects using YOLOv5 transfer learning in varying lighting and orientation. 2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology IEEE, pp. 1–7. DOI: 10.1109/IRASET52964.2022.9738370.
29. Jocher, G. et al. (2022). Ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation', DOI: 10.5281/ZENODO.7347926.
30. **Wang, C. Y., Bochkovskiy, A., Liao, H. Y. M. (2023).** YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 7464–7475.
31. Dorfer, T. A. (2023). Enhanced object detection: How to effectively implement YOLOv8. https://towardsdatascience.com/enhanced-object-detection-how-to-effectively-implement-yolov8-afd1bf6132ae.
32. **Terven, J., Cordova-Esparza, D. (2023).** A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. arXiv preprint arXiv:2304.00501.
33. **Rodriguez-Conde, I., Campos, C., Fdez-Riverola, F. (2022).** Optimized convolutional neural network architectures for efficient on-device vision-based object detection. Neural Computing and Applications, Vol. 34, No. 13, pp. 10469–10501. DOI: 10.1007/S00521-021-06830-W.

34. **He, K., Zhang, X., Ren, S., Sun, J. (2015).** Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, Vol. 37, No. 9, pp. 1904–1916. DOI: 10.1109/TPAMI.2015.2389824.
35. **Liu, S., Qi, L., Qin, H., Shi, J., Jia, J. (2018).** Path aggregation network for instance segmentation. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8759–8768. DOI: 10.1109/CVPR.2018.00913.
36. **Rainforest Expeditions (2022).** Guacamayo Azul y Amarillo.https://rainforestexpeditions.com/es/animales/guacamayo-azul-y-amarillo/.
37. **Cueto-Salazar, M. R. (2018).** Ararauna Tours and Expeditions. http://araraunatambopata.com/nosotros/.
38. **Chrome-stats** (2021). 'Fatkun Batch Descargar imagen. https://chrome-stats.com/d/nnjjahlikiabnchcpehcpkdeckfgnohf?hl=es.
39. **Seetala, K., Birdsong, W., Reddy, Y. B. (2019).** Image classification using tensorflow. 16th International Conference on Information Technology-New Generations ITNG 2019, Springer International Publishing, Vol. 800, pp. 485–488. DOI: 10.1007/978-3-030-14070-0_67.
40. **MINAM-Perú (2020).** Vive como sueñas. Reserva Nacional Tambopata, YouTube, https://www.youtube.com/watch?v=cDbkiWhs7SU&t=613s&ab_channel=MinisteriodelAmbiente-Perú.
41. **MINAM-Perú (2023).** Ministerio del ambiente. Plataforma del Estado Peruano, 2023. https://www.gob.pe/minam.
42. Antara-Perú (2023). Manu & Tambopata. YouTube, https://www.youtube.com/watch?v=hxoDoBH_7mA&t=3s&ab_channel= Antara Perú.
43. **Beleites, C., Salzer, R., Sergo, V. (2013).** Validation of soft classification models using partial class memberships: An extended concept of sensitivity & co. applied to grading of astrocytoma tissues. Chemometrics and Intelligent Laboratory Systems, Vol. 122, pp. 12–22. DOI: 10.1016/j.chemolab.2012.12.003.
44. **Palma-Ttito, L., Holgado-Apaza, L., Ccapa-Luque, R., Canaza-Canqui, E., Cornejo-Aparicio, V. (2021).** Detection of oligonucleotide microarray mutations by multiclass support vector machine. RISTI Revista Iberica de Sistemas e Tecnologias de Informacao, Vol. 2021, No. E39. pp. 643–657.
45. **Corrêa-Krüger, J. G., de-Souza-Britto-Jr, A., Barddal, J. P. (2023).** An explainable machine learning approach for student dropout prediction. Expert Systems with Applications, Vol. 233, p. 120933. DOI: 10.1016/J.ESWA.2023.120933.
46. **Kumar, A., Kalia, A., Kalia, A. (2022).** ETL-YOLO v4: A face mask detection algorithm in era of COVID-19 pandemic. Optik, Vol. 259, p. 169051. DOI: 10.1016/J.IJLEO.2022.169051.
47. **Suo, R., Gao, F., Zhou, Z., Fu, L., Song, Z., Dhupia, J., Cui, Y. (2021).** Improved multi-classes kiwifruit detection in orchard to avoid collisions during robotic picking. Computers and Electronics in Agriculture, Vol. 182, p. 106052. DOI: 10.1016/J.COMPAG.2021.106052.
48. **Fu, L., Yang, Z., Wu, F., Zou, X., Lin, J., Cao, Y., Duan, J. (2022).** YOLO-banana: A lightweight neural network for rapid detection of banana bunches and stalks in the natural environment. Agronomy, Vol. 12, No. 2. DOI: 10.3390/AGRONOMY12020391.
49. **Jocher, G. (2020).** YOLOv5 by ultralytics. DOI: 10.5281/zenodo.3908559.
50. **Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Wei, X. (2022).** YOLOv6: A single-stage object detection framework for industrial applications. arXiv preprint arXiv:2209.02976.
51. **Wang, C. Y., Bochkovskiy, A., Liao, H. Y. M. (2023).** YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7464–7475.
52. **Jocher, G., Chaurasia, A., Qiu, J. (2023).** YOLOv8 by Ultralytics. Ultralytics. https://github.com/ultralytics/ultralytics.

Article received on 11/10/2023; accepted on 11/03/2024.
**Corresponding author is Luis Alberto Holgado-Apaza.*

ISSN 2007-9737

Performance Analysis in Peer-to-Peer Networks with Collaborative Initial States

Gerardo Hernández-Oregón, Mario E. Rivero-Angeles*, Juan C. Chimal-Eguía, Jorge E. Coyac-Torres

Insituto Politécnico Nacional,
Centro de Investigación en Computación,
Mexico

{ghernandezo1101, jcoyact1900}@alumno.ipn.mx, {mriveroa, jchimale}@ipn.mx

Abstract. Currently, the trends of the new generations of telecommunications (6G and beyond) point towards a non-centralized process, however, for these architectures where no central entity regulates the transmission and information sharing, collaboration among nodes becomes a major issue to provide an adequate service. To this end, in this paper, we study Peer-to-Peer (P2P) systems to obtain models that reproduce these technologies' operation and the collaboration mechanism. The behavior of these systems is modeled through a Continuous Time Markov Chain (CTMC), and later we develop a Discrete Event Simulation (DES) to validate the analytical results. For solving these processes, M/M/s type queues were used in order to analyze the impact of collaboration levels on the system performance.

Keywords. Peer-to-peer (P2P) systems, continuous time Markov chain (CTMC), collaboration mechanism, discrete event simulation (DES), collaborative threshold (ζ).

1 Introduction

As a result of the creation of the first computer network, new topologies and data structures began to be implemented, in particular, centralized architectures where a server acts as the entity in charge of distributing information through the network. However, these architectures suffer from bottlenecks at the servers when popular files (and more recently videos and movies in streaming services) are downloaded and shared in social networks, producing high delays and degrading the performance of the networks.

In the same context, networks, such as Peer-to-Peer (P2P), emerge in a disruptive way to eliminate the need for central nodes and instead act together to complete tasks in common.

In a decentralized P2P network, a file is shared among the nodes that belong to the network, and all these nodes behave as servers and clients, sharing resources (memory, bandwidth, processing, and more).

In addition, in the case of the P2P-BitTorrent network studied in this article, the transmitted files are divided into small segments called chunks so that the nodes that contain the complete file are named seeds (x), while the Nodes that do not contain the full chunks are known as leechers (y).

Furthermore, while centralized networks aim to have distribution centers that can create conflicts of security, cost, stability, and other problems, P2P networks aim to expand to create reliable operations and make the nodes active in the network [7], however, in decentralized architectures apart from well-known security issues, such as worm distributions and viruses, the main problem is that the system operation relies heavily on nodes' collaboration to share their files and resources for the benefit of all users.

Although the phenomenon of collaboration has been largely overlooked through rules of connectivity, priorities and incentive mechanisms,

Table 1. P2P system states

State	Condition	τ
Penury	$c\,x_\sigma > \mu(\eta\,x_\sigma + y_\sigma)$	$\mu(\eta\,x_\sigma + y_\sigma)$
Abundance	$c\,x_\sigma < \mu(\eta\,x_\sigma + y_\sigma)$	$c\,x_\sigma$

this paper studies the fact that in real P2P networks, there are certain nodes that since the first moment connect anonymously or are not disposed to share information with the other peers but they also download the available chunks using the network's resources, for these reasons, it is important to highlight the fact that for our paper we consider an initial state of collaboration (σ) for every node in the network (υ_0, υ_1, υ_2, ..., υ_{n-1}, υ_n), where this σ state is a binary variable that says if the node is sharing information ($\sigma = \sigma_1 = 1$) or not ($\sigma = \sigma_0 = 0$). In the same line, we consider that a non-collaborative peer (υ_{σ_0}) is not willing to share information, even if it has all the information ready to share, on the contrary, a collaborative peer (υ_{σ_1}) will try to share data through all the network during its lifespan.

It is important to notice that every leecher and every seed contain a σ that defines their state of collaboration and for the case of conversions from leechers to seeds, the new seeds will preserve the antique σ state as when they were leechers. For the previous reasons, the total number of seeds can be expressed as $x_\sigma = x_{\sigma_1} + x_{\sigma_0}$, which includes the collaborative and non-collaborative seeds due to the initial σ state, as well, the total number of leechers is expressed as $y_\sigma = y_{\sigma_1} + y_{\sigma_0}$.

In this sense, a collaborative initial state (σ) allows greater fluidity in the processing of nodes for the transfer of information since it avoids the periods in which nodes have to determine priorities or apply incentive mechanisms in cooperation, allowing greater efficiency in nodes that require greater confidentiality without relegating them to the end of the process. Finally, a collaborative threshold (ζ) in the systems defines the maximum percentage of υ_{σ_0} nodes in the environment, and once this percentage was reached, this type of nodes could not connect or would have to change their policies to be added as a υ_{σ_1} or until some υ_{σ_0} nodes leave the system. The main contributions of this paper are listed below:

– A Markovian model is proposed and solved through a Continuous Time Markov Chain (CTMC) that describes the fluid system of a P2P system.
– The model is validated through chain solutions (fluid model and simulation), describing the steps to follow and comparing each solution with the fluid model.
– A variation in the simulation of the system is proposed in order to study the impact of collaboration for leechers and seeds evolution in time.
– The behavior of a network that has variation in its collaboration is described.

The rest of the article is composed as follows: Section 2, presents the technologies in which P2P networks have been applied, as well as studies on the collaboration of nodes in this type of networks.

Section 3 details the dynamics of the P2P networks studied in this article through their fluid model, while Section 4 models the system through a CTMC and presents the algorithms that solve this chain through its fluid model and its simulation.

The final part of the article shows the relevant results from the CTMC solution and the impact of collaboration through the variation of υ_{σ_0} nodes in the network, as well as conclusions and future work.

2 Related Work

The excessive growth in computing capabilities, the Internet, and new technologies such as IoT, AI, and VR are determining factors to search for other technologies and network architectures [8].

In the same line, P2P systems allow to implement decentralized file sharing by allowing all nodes to simultaneously act as servers and clients, increasing the resources in the network and, although P2P are still in constant development of cost reduction [14] in their infrastructure and detection of malicious nodes that may affect network processes [2], they are

Table 2. Solutions of the fluid model for stable conditions

State	Leechers	Seeds	γ
Abundance	$x_\sigma = \dfrac{\lambda}{\theta + c}$	$y_\sigma = \left(\dfrac{c}{\gamma}\right)\left(\dfrac{\lambda}{\theta + c}\right)$	$\gamma < \dfrac{\mu\, c}{c - \mu\eta}$
Penury	$x_\sigma = \dfrac{\lambda}{\theta + \mu\,\eta\left(1 - \dfrac{\mu}{\gamma - \mu}\right)}$	$y_\sigma = \dfrac{\mu\,\eta\, x_\sigma}{\gamma - \mu}$	$\gamma > \dfrac{\mu\, c}{c - \mu\eta}$

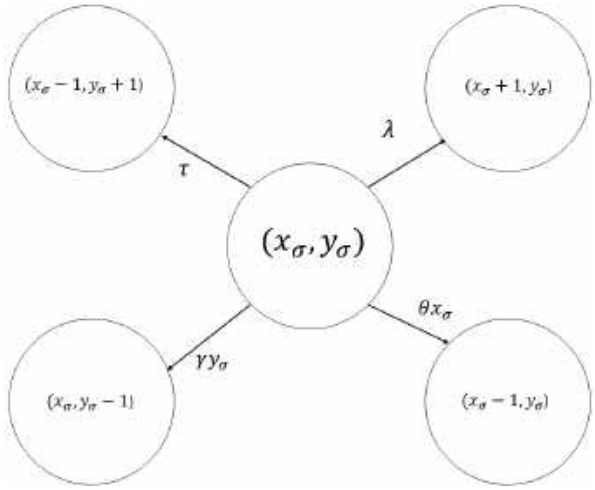


Fig. 1. CTMC of the P2P (BitTorrent) network

emerging as an alternative that could complement current environments having a significant peak in streaming services and with the future promise of a decentralized Internet as described by Philip Agre [1]. On the other hand, another important feature of P2P networks is the relationship that the peers maintain between them and the rules that they follow through the network.

In the same vein, few studies investigate the disposition of nodes to share information on the network and, although most refer to this process as collaboration, this collaboration can be approached from different points of view as shown by Chow [5, 4], who details how mobile nodes transmit information through caching techniques to reduce download times and to efficiently share information in the system, in the same way, Kumar [9] uses this procedure to vehicular networks by modeling a discrete-time Markov chain that describes the initial, waiting, and updating states of these scenarios and the structures necessary for their operation.

The previous works manage the collaboration through the structure in which the nodes share due to their mobility and their physical capabilities in the transmission of data, however, there are other works that consider collaboration in P2P networks in terms of exchange not only of information but of energy as proposed by Amin [3] in its P2P electricity network in which there are collaborative and non-collaborative houses depending if they have solar panels and if they share them with the network.

It is important to note that there are interesting works whose objective is to study the collaboration in order to obtain certain parameters of the

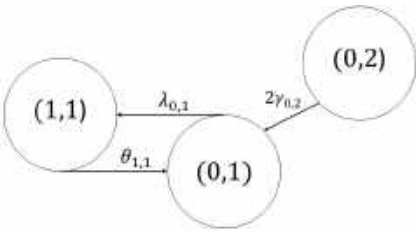


Fig. 2. Input and output transition rates for the state (0,1)

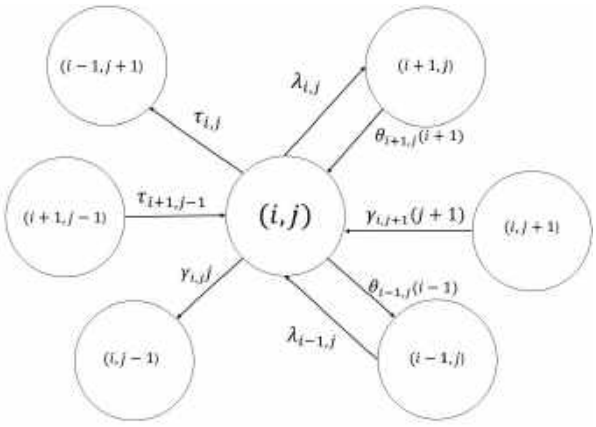


Fig. 3. Input and output transition rates for the state (i, j)

system, such as the one accomplished by Edgar Báez [6] for the calculation of collaborative and non-collaborative peers with different death rates, i.e., in this case the collaboration level is related to the average dwelling times inside the system, according to their priority scheme calculating latency, delay, successful downloads and percentage of blocked peers, or the study presented by Petrovic [11] in which a P2P system is analyzed in which there are different files of interest, then he details the download times and memory requirements for this type of scenario.

Distinctively to the articles described, this article focuses on the process of a Markovian analysis of the P2P system to study the collaboration among nodes considering that the peers enter anonymously (it is not possible to apply priority schemes) and determining if it will be collaborative (υ_{σ_1}) or non-collaborative (υ_{σ_0}) (through a proposed collaborative threshold (ζ)) from the first moment that the node enters in the network and keeping it in that state until its departure, finally, we analyze the impact of these initial states (σ_1, σ_2, σ_3, ..., σ_{n-1}, σ_n) of collaboration through the evolution of the network on the average number of leechers and seeds.

Those adjustments allow us to improve the speed of sharing information and safeguard nodes' anonymity, which means a better performance of P2P networks applied to technologies such as IoT that connect a lot of devices through the same network (many times in real-time) avoiding larger control information frames both uploading and downloading data.

3 P2P Networks

As we mentioned before, P2P-BitTorrent networks are very useful for sharing trending information through a large number of nodes with better performance than centralized networks, moreover, the P2P behavior is caused by some parameters that will be detailed in this section, however, it is just as important how nodes connect to each other according to the following intrinsic rules of P2P mechanisms:

– Seeds do not connect to each other, because they already contain all the information available in the network, however, they can connect to any leecher to upload parts of the file.

– Leechers that connect to any seed can download all available chunks (if network rules allow it).

– Leechers will only connect to other leechers that have different chunks between them.

Likewise, the nodes of the system can download with a rate c and upload information with a rate μ, having as a general (but not necessary) assumption:

$$c \geq \mu. \tag{1}$$

It is important to mention that bandwidths c and μ will determine the rate in which a leecher becomes a seed (τ), in such a way that, if the system has enough resources, it means that the leechers are interacting with a high upload rate of chunks and this generates a greater available bandwidth, which means that

Algorithm 1: Markov chain algorithm for a P2P network

Data: η, iterations, μ, c, nodes, λ, θ, γ
Result: x_σ, y_σ

```
tsim ← 0;
list.add(x_σ);
x_σ ← 1;
y_σ ← 0;
while i ≤iterations do
    tmin, position ← min(η, μ, λ, θ, γ);
    tsim ← tsim + tmin;
    if node ∃ in list then
        node.time ← node.time + tmin;
    else
        list.add(x_σ, node.time);
    switch position do
        case 0 do
            x_σ = x_σ + 1;
        case 1 do
            x_σ = x_σ − 1;
        case 2 do
            y_σ = y_σ − 1;
        otherwise do
            x_σ = x_σ − 1;
            y_σ = y_σ + 1;
list_pr ← probabilities(list);
for node in list do
    x_σ = x_σ + node.probability × node.leechers;
    y_σ = y_σ + node.probability × node.seeds;
```

the maximum download rate will be given by the product between the system leechers and the download rate ($c\,x_\sigma$).

Conversely, if the interaction of leechers is performed through a large number of repeated chunks (modeled by η), bottlenecks could appear indicating that all available bandwidth is being consumed and causing a lower download rate in the network, therefore, the conversion of lecheers to seeds will be modeled by the upload bandwidth, the number of seeds and the interaction between leechers ($\mu(\eta x_\sigma + y_\sigma)$). Finally, the rate at which leechers are converted to seeds is defined as τ and is given by the following relationship:

$$\tau = \min(c\,x_\sigma,\, \mu(\eta x_\sigma + y_\sigma)), \tag{2}$$

where
τ = conversion rate ($\forall \tau \in \mathbb{R}, \tau > 0$).
c = downloading bandwidth ($\forall c \in \mathbb{R}, c > 0$).
μ = uploading bandwidth ($\forall \mu \in \mathbb{R}, \mu > 0$).
η = sharing effectiveness ($\forall \eta \in \mathbb{R}, 0 < \eta <= 1$).
x_σ = number of leechers ($\forall x_\sigma \in \mathbb{N}, x_\sigma >= 0$).
y_σ = number of seeds ($\forall y_\sigma \in \mathbb{N}, y_\sigma >= 1$).

This system can be represented as a fluid model analyzed by D. Qiu [13, 12, 10] and developed for P2P-BitTorrent networks, and it can be expressed in a system of differential equations that are shown below in Equations 3 and 4:

$$\frac{d(x_{\sigma_1} + x_{\sigma_0})}{dt} = \lambda - \theta\,(x_{\sigma_1} + x_{\sigma_0}) - \tau, \tag{3}$$

$$\frac{d(y_{\sigma_1} + y_{\sigma_0})}{dt} = \tau - \gamma\,(y_{\sigma_1} + y_{\sigma_0}). \tag{4}$$

For stable conditions, in which a slight disturbance does not produce too disruptive effects on the system, we have:

$$\lambda - \theta x_\sigma - \tau = 0, \tag{5}$$

$$\tau - \gamma\, y_\sigma = 0. \tag{6}$$

Depending on the values of their bandwidths and output/input rates to the system, this type of network is classified into two states: penury and abundance. This relation is described in Table 1. For abundance state we substitute $\tau = c\,x_\sigma$ in equations 5 and 6:

$$\lambda - \theta\, x_\sigma - c\, x_\sigma = 0, \tag{7}$$

$$c\, x_\sigma - \gamma\, y_\sigma = 0. \tag{8}$$

Clearing x_σ variable:

$$x_\sigma = \frac{\lambda}{\theta + c}, \tag{9}$$

$$x_\sigma = \frac{\gamma\, y_\sigma}{c}. \tag{10}$$

Then:

$$\frac{\gamma y_\sigma}{c} = \frac{\lambda}{\theta + c}.$$

Solving y_σ:

$$y_\sigma = \left(\frac{c}{\gamma}\right)\left(\frac{\lambda}{\theta + c}\right).$$

The same way is applied to find x_σ and y_σ values for $\tau = \mu(\eta x_\sigma + y_\sigma)$ in penury state, the range of values that are valid for the departure rate of the seeds (γ) is obtained depending on the state in which the study is to be carried out, as shown in Table 2.

4 Mathematical Analysis

This paper studies the operation and collaboration of the P2P networks through a Markovian analysis in which the system can be represented as a homogeneous, irreducible, aperiodic CTMC with states (x_σ, y_σ) positive recurring (ergodic chain) like the one shown in Fig. 1, where:

λ = leechers input rate ($\forall\lambda \in \mathbb{R}$, $\lambda > 0$).
θ = leechers output rate ($\forall\theta \in \mathbb{R}$, $\theta > 0$).
γ = seeds output rate ($\forall\gamma \in \mathbb{R}$, $\gamma > 0$).
τ = conversion rate ($\forall\tau \in \mathbb{R}$, $\tau > 0$).
x_σ = number of leechers ($\forall x_\sigma \in \mathbb{N}$, $x_\sigma >= 0$).
y_σ = number of seeds ($\forall y_\sigma \in \mathbb{N}$, $y_\sigma >= 1$).

CTMC has a space of valid states $\Omega : \{(x_\sigma, y_\sigma) \mid x_\sigma, y_\sigma \in \mathbb{N}\}$, such for the states (x_σ, y_σ) it has the following transitions:

- There is a transition to state $(x_\sigma + 1,\ y_\sigma)$ with rate λ and denotes the arrival of a new leecher to the system.
- There is a transition to the $(x_\sigma - 1,\ y_\sigma)$ state with rate $(\theta)(x_\sigma)$ and denotes the output of a leecher from the system.
- There is a transition to the $(x_\sigma,\ y_\sigma - 1)$ state with rate $(\gamma)(y_\sigma)$ and denotes the output of a seed from the system.
- There is a transition to the $(x_\sigma - 1,\ y_\sigma + 1)$ state with rate (τ) and denotes the conversion from a leecher to a seed.

Algorithm 2: Simulation algorithm for a P2P network with collaboration variation

```
Data: chunks, threshold, η, iterations, μ, c, nodes, λ, θ, γ
Result: x_σ, y_σ
tsim ← 0;
list.add(y_σ(ID, status, connections, chunks, σ_1));
list.add(x_σ(ID, status, connections, chunks, σ_1));
y_σ ← 1;
y_σ1 = 1;
y_σ0 = 0;
x_σ ← 0;
x_σ1 = 1;
x_σ0 = 0;
while i ≤ iterations do
    event =← list.get_first();
    tsim ← event.time();
    c_σ1, c_σ0 ← conversions(tsim);
    y_σ1 ← y_σ1 + c_σ1;
    y_σ0 ← y_σ0 + c_σ0;
    x_σ1 ← y_σ1 − c_σ1;
    x_σ0 ← x_σ0 − c_σ0;
    if event.type == x_σ then
        if event.action == birth then
            x_σ = x_σ + 1;
            event.action = death;
            event.time = exponential(tsim, θ);
            search_tx(event, tsim);
            event.time = exponential(tsim, λ);
            if random(0, 1) ≤ ζ then
                list.add(x_σ(ID, ..., σ_1));
                x_σ1 ← x_σ1 + 1;
            else
                list.add(x_σ(ID, ..., σ_0));
                x_σ0 ← x_σ0 + 1;
        else
            x_σ = x_σ − 1;
            event.clearconnections();
            if event.collaborative == True then
                x_σ1 ← x_σ1 − 1;
            else
                x_σ0 ← x_σ0 − 1;
    else
        if y_σ > 1 then
            y_σ ← y_σ − 1;
            event.clearconnections();
            if event.collaborative == True then
                y_σ1 = y_σ1 − 1;
            else
                y_σ0 = y_σ0 − 1;
        else
            event.time = exponential(tsim, γ);
```

The initial state is $(0,\ 1)$, where there is no leechers asking for information and there is just one seed, for the above we cannot hope for a

Table 3. Table of parameters and values utilized

Parameter	Value
λ	[0.11, 0.12, 0.13, 0.14, 0.15]
θ	[0.001, 0.002, 0.003, 0.004, 0.005]
γ	[0.0009, 0.0012, 0.0015, 0.0018, 0.0021]
ζ	[0.2, 0.4, 0.6, 0.8, 1.0]
iterations	50,000
c	0.002
μ	0.00125

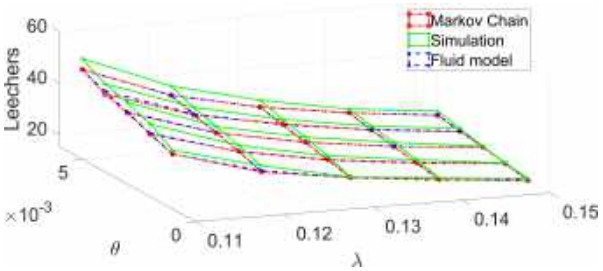


Fig. 4. Number of leechers in the system for the fluid model and the Markovian model in $\zeta = 1$

seed departure (because there must be at least one seed in the system) or a leecher departure but we can see that if state $(0,\ 2)$ presents a seed departure, we can return to $(0,\ 1)$ as is shown in Fig. 2. In the same way, we can generalized arrives and departures from state $(i,\ j)$ as in the Fig. 3 and for steady state this chain can be solved by equalizing their output and input transition rates, that is, by a flow balance Eqn. 11:

$$p_{0,1}(\lambda_{0,1}) = p_{1,1}\theta_{1,1} + 2p_{0,2}\gamma_{0,2}, \tag{11}$$

$$p_{1,1}(\lambda_{1,1} + \tau_{1,1} + \theta_{1,1}) = p_{0,1}\lambda_{0,1} + 2p_{2,1}\theta_{2,1}. \tag{12}$$

$$\vdots$$

$$p_{15,16}(\lambda_{15,16} + ... + \tau_{15,16}) = p_{14,16}\lambda_{14,16} + ... + \tau_{16,15}. \tag{13}$$

$$\vdots$$

$$p_{i,j}(\lambda_{i,j} + ... + \tau_{i,j}) = p_{i-1,j}\lambda_{i-1,j} + ... + \tau_{i+1,j-1}. \tag{14}$$

Once the Markovian model was described, the CTMC was solved numerically by modeling the jumps in the chain, creating exponential times corresponding to each transition rate as it is shown in Algorithm 1. After the numerical solution for the fluid model, we solve the chain by means of the Algorithm 2 simulating the events and characteristics of the system and applying the collaborative threshold ζ in the system.

Before any variation, our simulation was compared with the numerical solution to validate the data obtained in the simulation. A more detailed description of algorithms 1 and 2 is found below, in which, we take into consideration the collaboration between the nodes, remembering that for our paper we consider an initial state σ for every node in the network.

Furthermore, we remark and show the behavior of the state σ_i for a node i from the first moment that it gets into the network to the last moment it leaves the system. For the above, we remark that a υ_{σ_0} peer has an initial collaboration state $\sigma = 0$, which means that, this node will not share information, contrary, we consider a υ_{σ_1} peer when it has $\sigma = 1$.

4.1 CTMC Numerical Solution

The process of Algorithm 1 is detailed below:

– As a first step, the rates, bandwidths, and the number of iterations detailed in the Algorithm 1 are entered. However, for the simulation method, it will also be required to detail the number of chunks and the parameter ζ that will determine the collaboration between the present nodes in the network.

– In the next stage, the simulation time (tsim) and the number of leechers (x_σ) are initialized to zeros, and a seed is added to the list.

– From this moment on, the subsequent steps will be repeated during the indicated iterations. Next, the function $\min()$ is executed, which will create the exponential times corresponding to each possible state that can be accessed from the state $(x_\sigma,\ y_\sigma)$ and which will later determine the smallest time (tmin) and its position (position), taking into account that this shortest time corresponds to the time to the nearest event.

ISSN 2007-9737

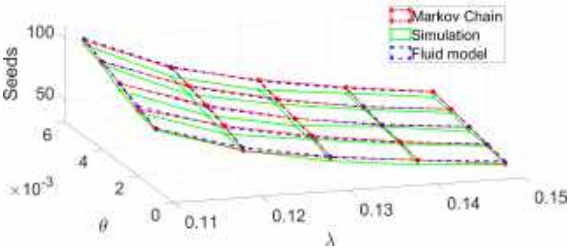


Fig. 5. Number of seeds in the system for the fluid model and the Markovian model with $\zeta = 1$

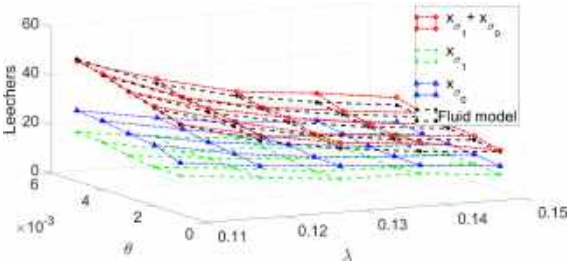


Fig. 6. Collaborative (x_{σ_1}) and non-collaborative (x_{σ_0}) leechers calculated by the simulations (ζ = 0.4, $\gamma = 0.0009$) compared to the ideal collaborative scenario (ζ = 1, $\gamma = 0.0009$)

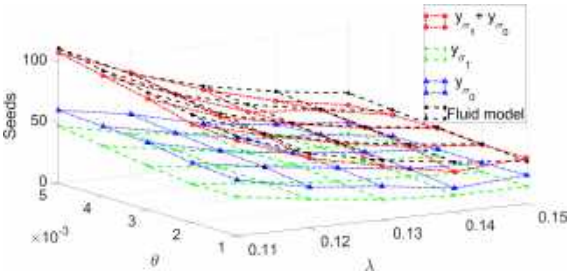


Fig. 7. Collaborative (y_{σ_1}) and non-collaborative (y_{σ_0}) seeds calculated by the simulations (ζ = 0.4, $\gamma = 0.0009$) compared to the ideal collaborative scenario (ζ = 1, $\gamma = 0.0009$)

– Once the minor time has been identified, it proceeds to add it to the simulation time, and this time would correspond to the time that the state $(x_\sigma,\ y_\sigma)$ remained before jumping to the new state.

– Subsequently, the transitions described above are considered cases in which the cases of conversion, departure of leechers/seeds from the system, and the arrival of leechers are considered.

– As a final step, the function probabilities() is executed, which will calculate the steady probabilities (list_pr) corresponding to each state present in the solution equations (15 and 16), to finally obtain the average number of leechers and seeds equations(17 and 18):

$$P(X_i) = \frac{\sum_{i=0}^{\infty} t(x_i)}{t_{\text{simulation}}}, \tag{15}$$

$$P(Y_i) = \frac{\sum_{i=0}^{\infty} t(y_i)}{t_{\text{simulation}}}, \tag{16}$$

$$P(\bar{X}) = \sum_{i=0}^{\infty} P(X_i)X_i, \tag{17}$$

$$P(\bar{Y}) = \sum_{i=0}^{\infty} P(Y_i)Y_i. \tag{18}$$

4.2 Discrete Event Simulator

As with the numerical solution algorithm, the corresponding steps of the algorithm are detailed by Algorithm 2:

– At first instance, the input rate of leechers (λ), the output rates of leechers and seeds (θ, γ), as well as the upload and download bandwidths (c, μ), the number of iterations (iterations), and the variable η are acquired. Finally, the average number of leechers and seeds in the system will be obtained.

– In the next stage, the simulation time (tsim) is initialized, then a seed and a leecher are added to the collaborative mode list. In other words, these nodes, in addition to downloading information, will collaborate by actively sharing them throughout the network. In order to have a count of the collaboration of the nodes, the counters of collaborative nodes (x_{σ_1}, y_{σ_1}) and non-collaborative nodes (x_{σ_0}, y_{σ_0}) are also initialized.

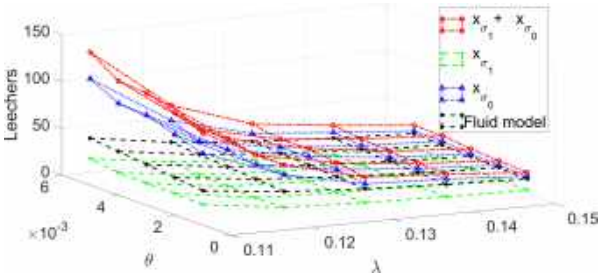


Fig. 8. Collaborative (x_{σ_1}) and non-collaborative (x_{σ_0}) leechers calculated by the simulation compared to the fluid model (ζ = 0.2, $\gamma = 0.0021$)

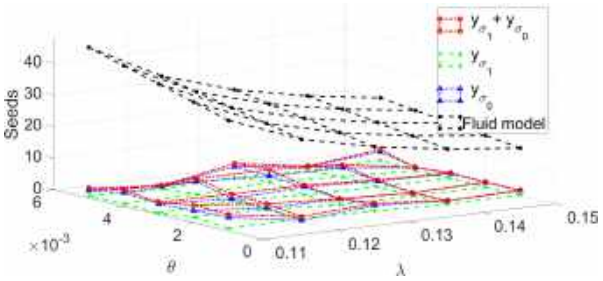


Fig. 9. Collaborative (y_{σ_1}) and non-collaborative (y_{σ_0}) $seeds$ calculated by the simulation compared to the fluid model (ζ = 0.2, $\gamma = 0.0021$)

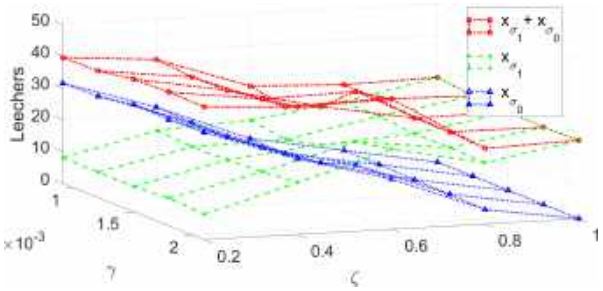


Fig. 10. Impact of the variation of ζ and the rate γ on the number of leechers of a P2P system (λ = 0.12, $\theta = 0.003$)

– The following steps will be repeated for the number of iterations entered. Next, the first node in the list (list.get_first()) will be taken, and its time (event.time()) will add that time to the simulation time.

– Subsequently, the number of conversions() from leechers to seeds in that time (if any) will be obtained and divided between collaborative (c_{σ_1}) and non-collaborative (c_{σ_0}) conversions.

– As the next step, the current event will be evaluated, and depending on its type and action to be carried out, following possibilities will be obtained:

  – If the type of the event (event.type) is a leecher and its action is a birth, then a unit is added to the leechers counter (collaborative or non-collaborative as the case may be) and a new time (event.time) is created for it, changing its action to death (event.action).

    Then, the possible providers of information in the network are searched for (search_tx()), and a new birth is created, which, depending on the random value (random(0, 1)) generated in comparison with the collaboration, this comparison determine if rebirth will be collaborative or non-collaborative.

  – If the type of the event is a leecher but has a death action, then the leecher's counter is decreased by one unit (x_{σ_1} or x_{σ_0}, as the case may be), and the upload and download connections of the node are eliminated (event.clearconnections()).

  – Finally, if the event type is a seed with a death action, then the seed counter is decreased by one unit (y_{σ_1} or y_{σ_0}, as the case may be), and the upload and download connections of the node are eliminated event.clearconnections(). It should be noted that there must always be at least one seed in the system, so if there is a death and only one seed in the system, a new death time for this node is generated.

– At the end of the iterations, the average number of leechers and seeds is obtained through the simulation, as well as the collaborative and non-collaborative nodes.

Under the collaborative circumstances described, we assume the same input (λ) and output (θ, γ) rates for collaborative peers and non-collaborative peers in order to observe the queuing of leechers and seeds when they are not willing to share information, this behavior is considered with the collaborative threshold ζ applied to the simulation when a peer is created, then, for every new node considered in the

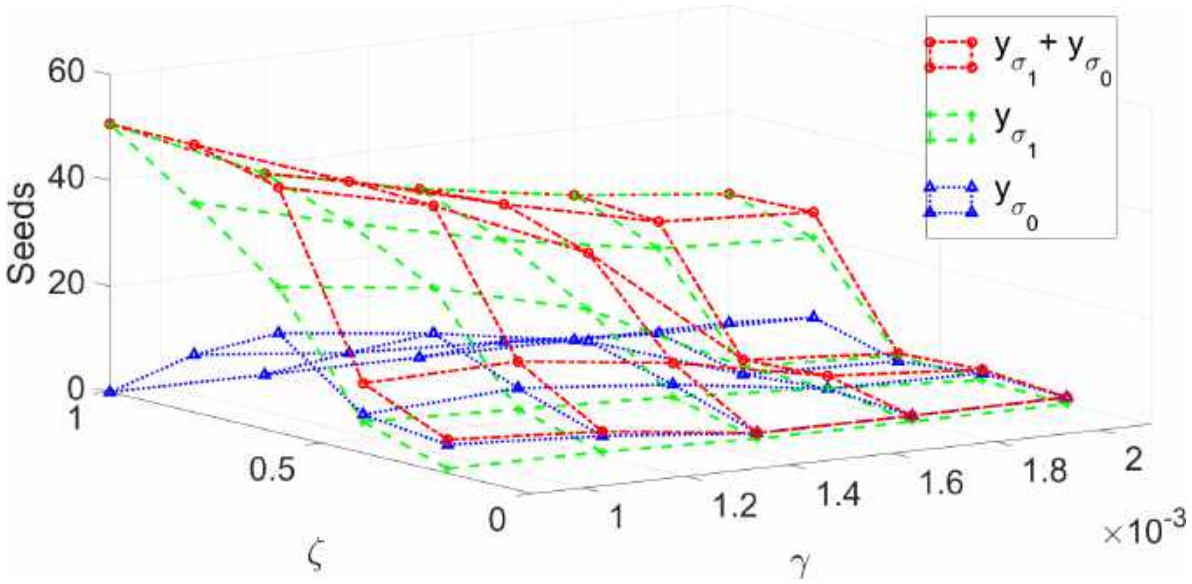


Fig. 11. Impact of the variation of ζ and the γ rate on the number of $seeds$ of a P2P system (λ = 0.12, $\theta = 0.003$)

simulation we decided the state σ by means of the comparison between a determined value of ζ ($0 < \zeta < 1$); if $\xi \leq \zeta$ then the new peer will have $\sigma = 1$ and it will be collaborative with the other nodes otherwise it will be non-collaborative with $\sigma = 0$. It is important to mention that when $\zeta = 1$ we have an ideal P2P environment in which all peers distribute information in the network, on the other hand, if $\zeta = 0$ we will have a centralized network in which only the initial node (that owns the file) could be capable of transmitting information to the other peers (Client-Serve architecture).

5 Results

Once the chain was solved, the fluid model was compared with the Markovian model through the DES of the system and the numerical solution of the Markov chain with $\zeta = 1$. The results of these comparisons are shown in the figures 4 and 5. At the same time, the parameters used for these solutions are indicated in Table 3. Figures 4 5 provide a validation of the proposed CTMC and the methods used for its solution. For this reason, the following subsection will show the results obtained from simulating the system with a collaborative threshold different from unity ($0 < \zeta < 1$).

5.1 Collaboration in P2P Environments

Ideally, all peers are willing to cooperate through the P2P network, however, in real environments, this could be different. The modeling referring to the P2P network studied in this work suggests another that allows glimpsing system's operation varying the ζ parameter in the network (Algorithm 1).

Figures 6 and 7 show that although the collaboration in this experiment was lowered, the values of leechers and seeds remained very close to the values calculated by the fluid model through the variation of the λ and θ rates with a γ value of 0.0009. The preceding shows that although many non-collaborative nodes exist in the system, an equilibrium is maintained that allows operation very close to the ideal operation.

However, figures 8 and 9 show a scenario in which collaboration between nodes is further reduced and output rate of the seeds (γ) increases, resulting in a network unable to provide the information in an effective way, producing a higher number of leechers in the queue and a very low number of seeds. Consequently, it can be affirmed that the level of collaboration directly impacts the conversion of leechers to seeds.

Hence, figures 8 and 9 result from variations in the arrival and departure rates of the nodes (λ and θ respectively) while the departure rate of the seeds (γ) was constant with the aim of observe the bearing of the network from the behavior of the leechers, then, it is important to study the system through the collaboration sweep varying γ and holding λ and θ constant.

For the above, Figures 10 and 11 show that the lower the collaboration, the greater the number of leechers in the queue (since leechers only depend on their arrival rate), and they will mostly be non-collaborative, which would cause a low number of seeds in the system (the number of conversions decreases). On the contrary, the greater the collaboration, the number of leechers and seeds will remain stable.

6 Conclusions and Future Work

This work developed a Markovian model that describes a P2P network, besides, we solve the model numerically and the simulation of the system, validating the solutions' results with pre-existing fluid models.

On the other hand, necessary adjustments were made to achieve variation in collaboration between the nodes of the network through the parameter ζ, taking into account real environments in which it is not possible to control all the peers and their collaboration. Furthermore, we propose to include the initial state of collaboration σ in the simulation to provide a better approach to real environments and then estimate the performance in the network according to this control variable.

On the other hand, despite the adjustments applied allowing nodes to hold their anonymity, the parameter ζ defines the total percentage of σ_0 nodes authorized in the network, corroborating that the smaller number of x_{σ_0} nodes in the environment the greater number of y_{σ_1} are created and then, the complete file can be shared through all the nodes more efficiently.

For the above, it can be affirmed that a P2P network can work efficiently, finding the balance of its rates even if its collaboration is not maximum ($\zeta = 1$) and thus maintaining a performance similar to that presented by the fluid model.

However, it is clear that if the balance as mentioned above is not achieved, a null creation of collaborative leechers (x_{σ_1}) can be obtained, which in turn translates into a very low number of seeds and, therefore, an inefficient P2P network in which it is complicated to get the information within the network. Therefore, it would be convenient and interesting to calculate the collaboration balance for future work and determine the impact found through ζ directly in the fluid model.

Data Availability

All experiments and data used in this work are available if required.

Acknowledgments

This research was funded by Instituto Politécnico Nacional grant number SIP 20231239 and CONACYT.

References

1. **Agre, P. E. (2003).** P2P and the promise of internet equality. Communications of the ACM, Vol. 46, No. 2, pp. 39–42. DOI: 10.1145/606272.606298.
2. **Al-Otaiby, N., Alhindi, A., Kurdi, H. (2022).** Anttrust: An ant-inspired trust management system for peer-to-peer networks. Sensors, Vol. 22, No. 2, pp. 533. DOI: 10.3390/s22020533.
3. **Amin, W., Huang, Q., Afzal, M., Khan, A. A., Umer, K., Ahmed, S. A. (2020).** A converging non-cooperative and cooperative game theory approach for stabilizing peer-to-peer electricity trading. Electric Power Systems Research, Vol. 183, pp. 106278. DOI: 10.1016/j.epsr.2020.106278.
4. **Chi-Yin, C., Hong-Va, L., Chan, A. T. S. (2007).** GroCoca: Group-based peer-to-peer cooperative caching in mobile environment. IEEE Journal on Selected Areas in Communications, Vol. 25, No. 1, pp. 179–191. DOI: 10.1109/jsac.2007.070118.

5. **Chow, C. Y., Leong, H. V., Chan, A. (2004).** Peer-to-peer cooperative caching in mobile environments. Proceedings of the 24th International Conference on Distributed Computing Systems Workshops, pp. 528–533. DOI: 10.1109/icdcsw.2004.1284083.
6. **Esquivel, E. E. B., Rivero-Angeles, M. E., Orea-Flores, I. Y. (2014).** Performance analysis of bittorrent-like P2P networks for video streaming services at the chunk level. Proceedings of the IEEE 28th International Conference on Advanced Information Networking and Applications, pp. 806–812. DOI: 10.1109/AINA.2014.98.
7. **Karaata, M., Al-Mutairi, A., Alsubaihi, S. (2022).** Multipath routing over star overlays for quality of service enhancement in hybrid content distribution peer-to-peer networks. IEEE Access, Vol. 10, pp. 7042–7058. DOI: 10 .1109/access.2021.3139936.
8. **Kumar, D., Pandey, M. (2022).** An optimal load balancing strategy for P2P network using chicken swarm optimization. Peer-to-Peer Networking and Applications, Vol. 15, No. 1, pp. 666–688. DOI: 10.1007/s12083-021-012 59-3.
9. **Kumar, N., Lee, J. H. (2014).** Peer-to-peer cooperative caching for data dissemination in urban vehicular communications. IEEE Systems Journal, Vol. 8, No. 4, pp. 1136–1144. DOI: 10.1109/jsyst.2013.2285611.
10. **Ma, Z., Qiu, D. (2009).** A novel optimistic unchoking algorithm for bittorrent. Proceedings of the 6th IEEE Consumer Communications and Networking Conference, pp. 1–4. DOI: 10.1109/CCNC.2009.4784859.
11. **Petrovic, S., Brown, P. (2009).** Large scale analysis of the eDonkey P2P file sharing system. Proceedings of the IEEE International Conference on Computer Communications, pp. 2746–2750. DOI: 10.1109/infcom.2009. 5062224.
12. **Qiu, D., Sang, W. (2008).** Global stability of peer-to-peer file sharing systems. Computer Communications, Vol. 31, No. 2, pp. 212–219. DOI: 10.1016/j.comcom.2007.08.012.
13. **Qiu, D., Srikant, R. (2004).** Modeling and performance analysis of bittorrent-like peer-to-peer networks. ACM SIGCOMM Computer Communication Review, Vol. 34, No. 4, pp. 367–378. DOI: 10.1145/1030194. 1015508.
14. **Singh, S. K., Kumar, C., Nath, P. (2022).** Resource-cardinality based scheme to reduce resource lookup cost in structured P2P networks. Wireless Personal Communications, Vol. 125, No. 4, pp. 3351–3377. DOI: 10.1007/ s11277-022-09714-x.

Article received on 16/10/2023; accepted on 21/11/2023.
** Corresponding author is Mario E. Rivero-Angeles.*

ISSN 2007-9737

Policy Gradient MaxSAT Solver

Omar Gutierrez-De-La-Paz, Ricardo Menchaca-Méndez*, Erik Zamora-Gómez, Uriel Corona-Bermudez, Rolando Menchaca-Méndez, Bruno Gutiérrez-De-La-Paz

Instituto Politécnico Nacional,
Centro de Investigación en Computación,
Mexico

{ogutierrezd2019, ric, rmen, bgutierrezd2019}@cic.ipn.mx, {ezamorag, ucoronab}@ipn.mx

Abstract. This paper presents a comparative study of various elements and strategies that can be incorporated into an autoregressive model to address the MaxSAT problem. Building upon a sequential architecture as our foundation, we optimize the model's parameters by maximizing the expected number of satisfied clauses. This optimization enables the model, given a SAT formula, to predict a distribution over potential solutions using the policy gradient method. Our controlled experiments pinpoint elements that guide the optimization process towards superior results[1].

Keywords. MaxSAT problem, policy gradient, NP-hard.

1 Introduction

The Maximum Satisfiability Problem (MaxSAT) is an optimization variant of the Boolean Satisfiability Problem (SAT). Its goal is to identify a truth assignment that maximizes the number of satisfied clauses in a given boolean formula. MaxSAT represents a classical challenge in computational theory and has wide-ranging applications across numerous domains due to its generic representation of optimization problems [10, 16, 24, 11]. While recent advances in machine learning, especially deep learning, provide promising approaches to combinatorial problems [5, 4, 25, 19], the adaptation of these techniques to the MaxSAT problem is still an emerging area of study.

This research delves into the comparative analysis of various elements and strategies, all aimed at enhancing the efficiency of an autoregressive model tailored to address the MaxSAT problem. Central to our exploration is the role of sequential models that capture current and past relevant information to generate convenient variables' values, baseline methods used to mitigate variance during optimization, and the technique used to represent variables and incorporate semantic information.

By systematically examining these components through our controlled hyperparameter searches across a diverse set of random 3-SAT instances, we aim to figure out the contributions of each element to the solution quality. The emphasis is on discerning the configurations that consistently yield superior results.

A crucial aspect of our methodology is the employed optimization technique. We leverage the policy gradient theorem, optimizing the parameters of the model on individual MaxSAT instances by maximizing the expected number of satisfied clauses to learn a distribution over potential solutions. This process dynamically adjusts the probability associated with variables taking specific values, true or false, as illustrated in fig. 1. It is crucial to underscore, however, that the ambition of this paper isn't to compete with the state-of-the-art solvers such as the MaxHS [9, 3], Open-WBO [22, 23], EvalMaxSAT [2], or Loandra [6].

Instead, our goal is centered on discovering strategies and elements that enhance the effectiveness of Deep Learning approaches.The remainder of the paper is organized as follows. In Section 2, we review recent approaches to

[1] https://github.com/omargup/Policy-Gradient-MaxSAT-Solver

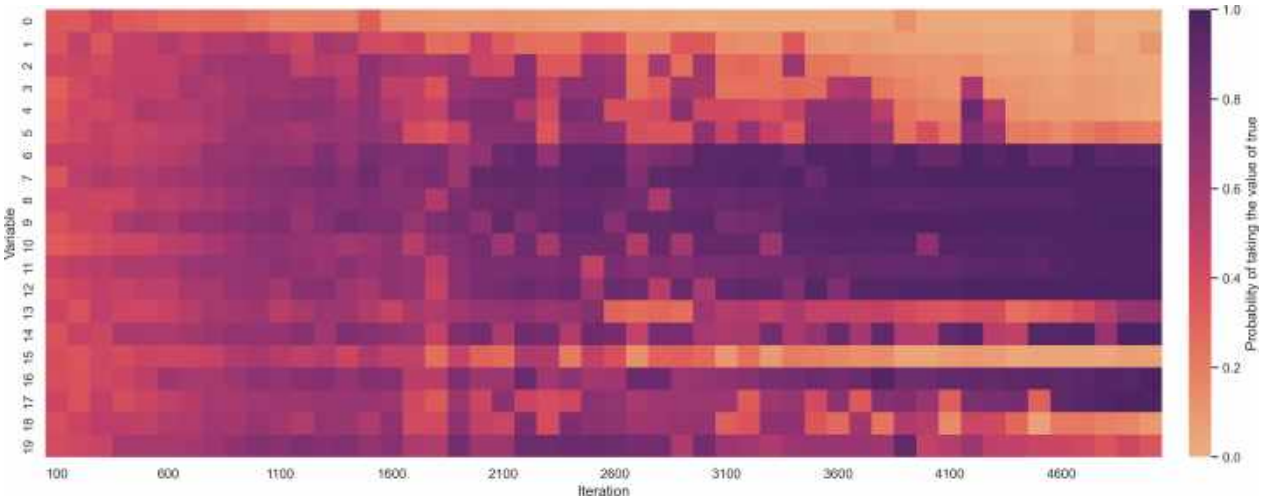


Fig. 1. Evolution of the probability that variables assume the value of true, $p(x_i = 1)$, over 5000 iterations for a random 3-SAT instance with 20 variables and 90 clauses. The ith row showcases the changes in the probability of variable x_i being true throughout the optimization process. Purple indicates a high probability of the value being true, whereas yellow denotes a lower probability (i.e., a higher likelihood of the value being false). At the beginning, there is uncertainty regarding the values the variables will assume; however, as the process unfolds, the probabilities converge towards either one or zero

solving the MaxSAT and related problems with Deep Learning. Subsequently, in Section 3, we detail the problem, the optimization process, and the configurable components of our model. In Section 4, we outline the proposed experiments and, finally, in Section 5, we discuss our findings.

2 Related Work

A variety of Machine Learning-based methods for combinatorial optimization problems have been developed that construct solutions sequentially through Reinforcement Learning [4, 17, 19]. These methods are well-suited for integration with search strategies like sampling and beam search, providing strong guidance in the search process. Bello et al. [4] introduced a generic search strategy known as active search. This strategy allows for guided exploration in solution construction without the need for problem-specific components. Unlike traditional methods that only sample solutions at inference time with a fixed model, active search operates iteratively on a single test input, modifying the model's parameters to increase the likelihood of generating high-quality solutions in future iterations.

Their focus was predominantly on the Traveling Salesman Problem (TSP). In their study, Bello et al. compared two methodologies of active search. The first approach involves adjusting the weights of a pre-trained model specific to a test instance, utilizing Reinforcement Learning. The second approach initiates active search on a single instance with an untrained model.

Their findings showed enhanced performance compared to random sampling when starting the search with a pre-trained model. Notably, initiating the search with an untrained model also led to satisfactory results. Hottung et al. [15] developed and evaluated three efficient active search strategies that update only a selected subset of parameters during the search. This approach contrasts with the full-scale model weight updates implemented in [4].

Their results indicated significant enhancements in search performance of models, outperforming some of the leading Machine Learning-based methods in various combinatorial problems, such as the TSP, Capacitated Vehicle

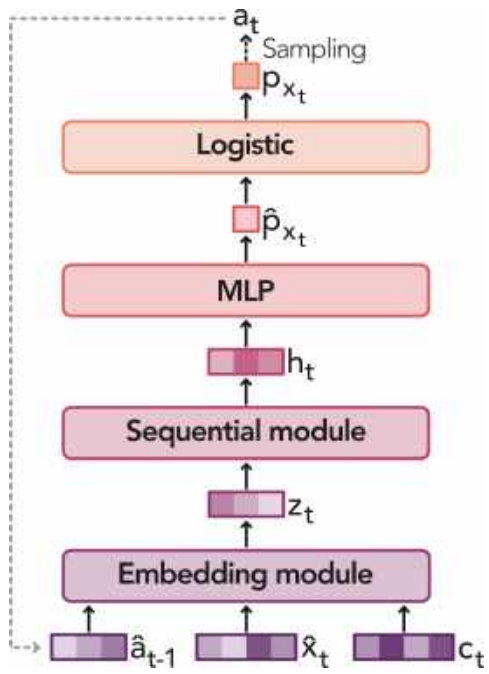


Fig. 2. High-level description of the model. At each time step t, the model ingests a representation $\hat{x}_t$ of the current variable, a representation $\hat{a}_{t-1}$ of the previous variable's value, and the context c_t. It then produces a 1-dimensional output p_{x_t} in the range [0,1] which is interpreted as the probability that the variable's value is true. Subsequently, the truth value a_t for x_t is sampled from p_{x_t}

Routing Problem (CVRP), and Job Shop Scheduling Problem (JSSP). Our research explores the latter approach in [4], particularly focusing on active search using an untrained model for a single input.

We aim to exploit the capabilities of neural networks, gradient descent, and Reinforcement Learning as distribution-independent optimization tools to find efficient solutions for the MaxSAT problem by transitioning the optimization approach from a discrete to a continuous domain. While Hottung et al. focused on improving active search by updating a subset of parameters starting from a trained model, our research seeks to enhance active search beginning with an untrained model. We explore a range of architectural and optimization elements. For example, while Bello et al. used an exponential moving average (EMA) baseline, we extend our exploration to include various sequential models, multiple baselines (including EMA), and the benefits of incorporating problem-specific information into the input via Node2Vec.

3 Methods

3.1 Problem Definition

A boolean variable x_i can assume values true (one) or false (zero). A literal l_i refers to the variable x_i or its negation $\neg x_i$. A clause c is represented as a disjunction (logical OR) of literals. A Conjunctive Normal Form (CNF) formula ϕ is expressed as a conjunction (logical AND) of clauses.

The SAT problem requires determining a truth assignment π that provides a truth value (true or false) to every variable in a CNF formula ϕ, such that all clauses are satisfied, or stating that no such assignment exists. The MaxSAT problem is an optimization variant of SAT that seeks an assignment which maximizes the number of satisfied clauses in ϕ.

Thus, even if not all clauses in the formula can be simultaneously satisfied, MaxSAT will identify an assignment that satisfies the maximum possible number of clauses. In this work, our primary focus is on the MaxSAT problem. Formally, given a CNF formula ϕ containing n variables and m clauses, our objective is to identify a truth assignment π maximizing the number of satisfied clauses, represented as $S(\pi|\phi)$.

3.2 Objective Function and Optimization Procedure

We employ an autoregressive model to approximate optimal solutions. Let $S(\pi^*|\phi)$ be the optimal number of satisfiable clauses, and $p_\theta(\pi|\phi)$ be a multi-dimensional distribution with learnable parameters θ.

Our aim is to approximate $S(\pi^*|\phi)$ by maximizing the expected number of satisfiable

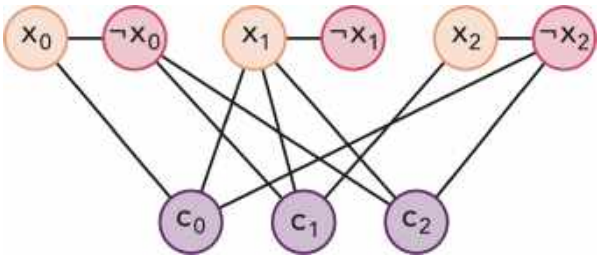


Fig. 3. Graph representation of the formula for the Node2Vec algorithm. The illustration corresponds to the formula $(x_0 \vee x_1 \vee \neg x_2) \wedge (\neg x_0 \vee x_1 \vee x_2) \wedge (\neg x_0 \vee x_1 \vee \neg x_2)$. Nodes symbolize literals and clauses. Undirected edges link literals to the clauses they appear in, while supplementary edges connect literals of the same variable

clauses $S(\pi|\phi)$ under the distribution $p_\theta(\cdot|\phi)$ represented by the model, i.e.:

$$S(\pi^*|\phi) \approx \max_\theta \mathbf{E}_{\pi\sim p_\theta(\cdot|\phi)} S(\pi|\phi), \tag{1}$$

where

$$J(\theta|\phi) = \mathbf{E}_{\pi\sim p_\theta(\cdot|\phi)} S(\pi|\phi). \tag{2}$$

Is the objective function.

To maximize this objective function for a specific instance of the MaxSAT problem, we optimize the model's parameters using stochastic gradient ascent using the ADAM optimizer [18], leveraging the policy gradient theorem. The gradient of $J(\theta|\phi)$ with respect to θ is given by:

$$\nabla_\theta J(\theta|\phi) = \mathbf{E}_{\pi\sim p_\theta(\cdot|\phi)} \left[S(\pi|\phi) \nabla_\theta \log p_\theta(\pi|\phi) \right], \tag{3}$$

where, due to the autoregressive nature of the model, the probability of an assignment can be factorized using the chain rule:

$$p_\theta(\pi|\phi) = \prod_{i=1}^{n} p_\theta(\pi_i|\pi_{i-1}, \pi_{i-2}, \ldots, \pi_1). \tag{4}$$

And hence, we can approximate the gradient by sampling assignments π from the model.

3.3 Architecture

To address the MaxSAT problem, we employ a sequential model. This model accepts as inputs a representation of the formula's variables and another representation of the formula itself, termed as context. Sequentially, for each variable, it produces an output probability from which the truth value of that variable is sampled.

The design of the model is autoregressive [12], ensuring that each step takes into account a representation of the previously generated output when generating the probability for the next variable's truth value.

In a high-level overview, the functioning of the model can be described as follows (refer to fig. 2): for a given time t in the range [0, $n-1$], given a sequence of variables' representation $\hat{\mathbf{x}} = (\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{n-1})$, the embedding module processes the current variable's representation $\hat{x}_t$ along with the context c_t and a representation of the previous assignment $\hat{a}_{t-1}$ to yield a vector z_t. Subsequently, the sequential model transforms z_t into a state h_t taking into account information from previous time steps.

Finally, a Multi-Layer Perceptron (MLP) processes h_t to produce a 1-dimensional output $\hat{p}_{x_t}$. We then map $\hat{p}_{x_t}$ to the range [0,1] using the logistic function, interpreting p_{x_t} as the probability of the variable's value being true. A truth value a_t for x_t is sampled from p_{x_t} considering 0 and 1 as false and true, respectively.

We initialize $a_0 = 2$ as our start of sequence (sos) token, and $\hat{a}_t$ is the corresponding 3-dimensional One-Hot encoded vector of a_t. We introduce several adjustable components in the architecture and the training methodology.

This is done to pinpoint features that contribute to a better approximation of the solutions, such as the sequential model, baseline methods, and variables' representations. These components are elaborated further in the subsequent sections.

3.3.1 Variables Representations and Context

To underscore the importance of integrating problem instance information into the representation of variables when addressing the MaxSAT problem, we delve into two distinct methods: One-Hot encoding and Node2Vec [13] embeddings.

The former offers a direct, albeit potentially limited, approach that signifies

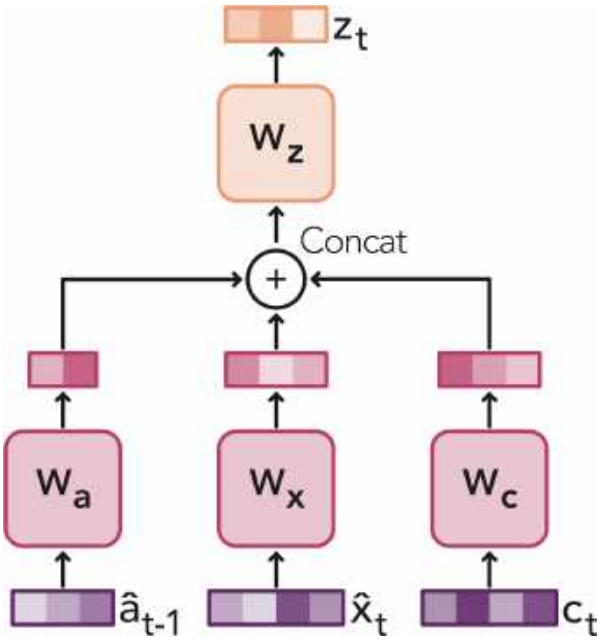


Fig. 4. Embedding module. The module intakes three inputs: the previous variable value representation $\hat{a}_{t-1}$, the current variable representation $\hat{x}_t$, and the context c_t. Each input is linearly transformed into its respective embedded vector with dimensions determined by configurable hyperparameters. These vectors are then concatenated to form a unified representation, encapsulating the essence of the initial inputs. This unified structure is further linearly transformed to generate the resultant vector z_t, defined by a designated hyperparameter dimension

the index of variables without encapsulating formula-specific information. On the other hand, Node2Vec is an algorithmic framework devised for learning continuous feature representations of nodes within networks.

It adopts a graph-based approach that seamlessly blends the strengths of structural equivalence (nodes sharing similar roles) with homophily (nodes associating with similar neighbors), thereby crafting versatile node embeddings. Building on this, our adaptation of the Node2Vec methodology entails crafting a graph representation of the formula, drawing inspiration from [25]. In this graph, nodes correspond to literals and clauses.

Undirected edges link literals and clauses that interrelate within the formula, and supplementary edges connect literals belonging to the same variable (see fig. 3). This comprehensive graph approach allows Node2Vec to encode formula-specific insights, potentially enhancing the optimization processes and solution quality.

While both methods provide unique perspectives, Node2Vec delves deeper into the instance's structure, though it necessitates a separate learning phase to acquire the feature representations of the nodes. For the Node2Vec variables' representation, we construct a vector for a specific variable by concatenating the Node2Vec embeddings of its corresponding two literals. This manner of representation captures insights concerning both literals of the variable.

Regarding the context, we average the embeddings corresponding to the literals, and similarly, average the embeddings linked to the clauses. The resultant context vector is the concatenation of these two averaged vectors, ensuring that the context includes comprehensive information about the instance, which could be invaluable when attempting to solve the problem.

3.3.2 Sequential Module

In our study, we compare and analyze three distinct models within the sequential module of the architecture: a Transformer encoder [26], an Long Short-Term Memory (LSTM) [14] network and Gated Recurrent Units (GRU) [8].

Both recurrent neural networks (RNNs) and the Transformer offer unique characteristics and capabilities for approximating solutions to the MaxSAT problem.

The Transformer encoder leverages self-attention mechanisms to capture dependencies among variables and effectively model long-range dependencies. In contrast, the LSTM and GRU models, also capture sequential information and dependencies over time but typically exhibit better memory efficiency when compared to Transformers. By examining and comparing these three models, we aim to assess their performance and determine the most

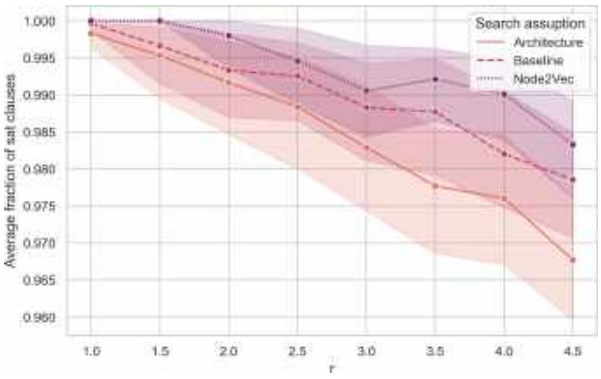


Fig. 5. Average fraction of satisfied clauses as the radius r increases. The average encompasses the six different values of n and their respective five different instances

effective approach for approximating MaxSAT solutions within our proposed architecture.

3.3.3 Embedding Module

To ensure robustness in our model and adaptability across different experiments, there was a need to address potential compatibility and expressiveness issues that arise when altering various components of the model. An example is the variability in representation sizes of the variables. For instance, while One-Hot encoding has an n-dimensional representation, the Node2Vec embedding's size is determined by a hyperparameter.

Additionally, unlike recursive networks that accommodate input vectors of any dimension, the Transformer architecture necessitates that the input dimension aligns with the model's dimension. To navigate these constraints and ensure a harmonious flow in the architecture, we introduced an embedding module.

This module was conceptualized to bridge the gap, linking the diverse inputs to the sequential module in a seamless and expressive manner. The core functionality of the embedding module is centered around transforming the three inputs, namely the previous variable value representation $\hat{a}_{t-1}$, the current variable representation $\hat{x}_t$, and the context c_t.

Each of these inputs undergoes a linear transformation to produce their respective embedded vectors, the dimensions of which are guided by configurable hyperparameters. This flexibility in defining dimensions allows the model to adapt effectively across a variety of experimental setups.

After these transformations, the three embedded vectors are concatenated to create a unified representation that captures the essence of all initial inputs while maintaining structural consistency.

This concatenated vector is subsequently linearly transformed to produce z_t, a resultant vector with a dimension defined by a specific hyperparameter. This vector serves as the input for the sequential module, ensuring it receives information in a consistent and streamlined format (see fig. 4).

3.3.4 Baselines

The policy gradient theorem can be generalized to include a baseline. Within our framework, a baseline $b(\phi)$ can be any function as long as it does not vary with the assignment π. It serves as a reference value (or baseline) against which the number of satisfied clauses from a particular assignment is compared.

Utilizing a baseline reduces the variance of the gradient estimate, which in turn stabilizes and accelerates the learning process. When incorporating a baseline $b(\phi)$, the equation (3) is adjusted reflecting the formulation of the REINFORCE [27] algorithm:

$$\nabla_\theta J(\theta|\phi) = \mathbf{E}\left[\delta \cdot \nabla_\theta \log p_\theta(\pi|\phi)\right], \qquad (5)$$

where the expectation is drawn from the distribution $p_\theta(\cdot|\phi)$ and $\delta = S(\pi|\phi) - b(\phi)$.

Acknowledging the importance of incorporating a suitable baseline, we experimented with three distinct baseline methods, in addition to the approach that operates without a baseline. Each of these methods aims to provide an estimate of the expected number of satisfied clauses:

– **Exponential Moving Average (EMA)**. This technique employs the exponential moving average to track the number of clauses satisfied by the model's assignment over time.

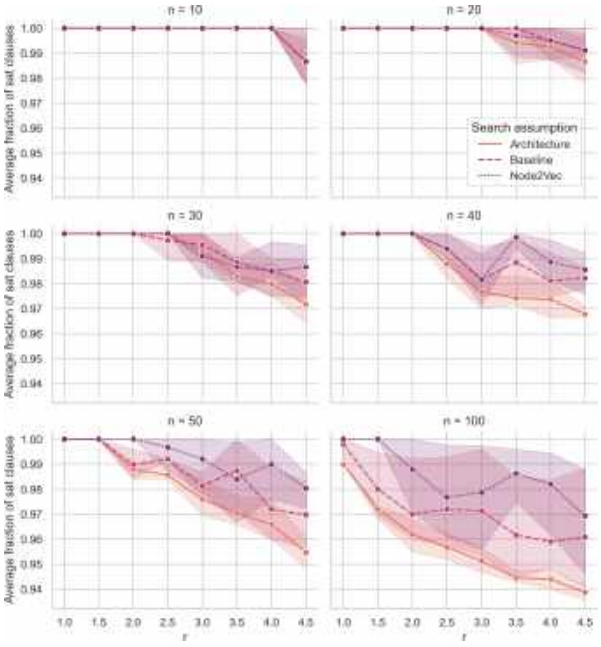


Fig. 6. Average fraction of satisfied clauses as the radius r increases for n =10, 20, 30, 40, 50, and 100. The average spans the five different instances for each n, and the plots are sequenced from left to right and top to bottom

– **Greedy Baseline**. With the current model parameters held constant, this baseline selects, at each time step, the most probable value for the current variable. It then calculates the number of clauses satisfied by the assignment.

– **Sampling Baseline**. In contrast to the greedy approach, this method draws B solutions and sets the baseline value as the average number of clauses satisfied by the drawn samples.

3.3.5 Exploration

To foster exploration within the model, and inspired by [4], we integrate two distinct strategies.

– **Logit Temperature**. At each time step t, the model's output is adjusted as:

$$\hat{p}_{x_t} = \frac{\hat{p}_{x_t}}{T}, \tag{6}$$

where T represents a temperature hyperparameter. During training, T is set to 1. However, during evaluation, when $T > 1$, the output $\hat{p}_{x_t}$ becomes less pronounced, thereby inhibiting the model from exuding overconfidence.

– **Logit Clipping**. At each time step t, the model's output undergoes another modification:

$$\hat{p}_{x_t} = C \cdot \tanh(\hat{p}_{x_t}), \tag{7}$$

where C serves as a hyperparameter, dictating the range of the logits and subsequently the entropy of the resultant output.

4 Experimental Setup

We conducted a series of experiments to assess the potential of a sequential model for approximating solutions to the MaxSAT problem. Our goal was to discern the architectural and procedural elements that enhance the optimization process to maximize the number of satisfied clauses.

Unlike the traditional SAT solvers, where the aim is to find a complete assignment that satisfies all clauses or determine its non-existence, our focus was on obtaining solutions that closely resemble optimality.

In our study, we carefully examined three pivotal elements: the decoder architecture, the baseline used to mitigate variance during optimization, and the embedding technique used to represent variables and incorporate semantic information. We posited that these components significantly influence the performance of the autoregressive model and, consequently, the quality of the MaxSAT solution.

To rigorously evaluate the contributions of these elements, we organized our experiments into three stages of hyperparameter searches. In the initial stage, the spotlight was on different decoder architectures, using a fixed zero baseline and rudimentary variables' representation to isolate the effect of the decoder structure on performance.

The subsequent stage introduced an extra layer of complexity by incorporating the various baselines into the hyperparameter search. Lastly,

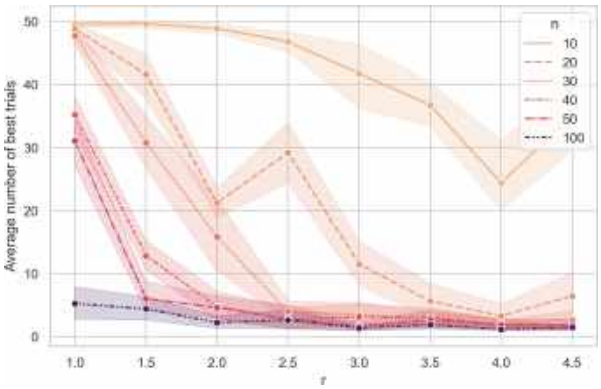


Fig. 7. Average number of best trials. The average is over the five different instances and the three different search assumptions

the Node2Vec embedding method was integrated into the final stage, broadening the search parameters and allowing for a comprehensive assessment of these elements' combined influence on model behavior.

Throughout these stages, we integrated several other hyperparameters into our searches, such as learning rate, number of layers, embedding dimensionality, logit clipping, logit temperature, and more. It is worth noting that crafting Node2Vec representations involves its own intricate procedure, detached from the parameter tweaking of the sequential model.

Consequently, we executed an independent hyperparameter search specifically for the Node2Vec embedding process, ensuring the availability of high-quality embeddings when required. By systematically varying these elements and exploring their interactions through the hyperparameter searches, our intention was to pinpoint the most efficient configurations for the autoregressive model in addressing the MaxSAT problem. This methodology enabled a examination of each component's individual contributions and their synergistic effects, steering the optimization process towards near-optimal solutions. Furthermore, to bolster the rigor and impartiality of our assessments, our experimental setup was tested across a diverse set of random 3-SAT instances.

4.1 Dataset

We generate a random k-SAT dataset comprising not necessarily satisfiable random formulas. For producing a random k-SAT instance with n variables and m clauses, we begin by choosing a small integer for k. Subsequently, for each clause c_i, where $i \in \{1, 2, \ldots, m\}$, we sample k variables uniformly at random without replacement. Each variable is then negated with a 50% independent probability.

The dataset encompasses 5 instances for each of the following configurations: for $n \in \{10, 20, 30, 40, 50, 100\}$, formulas with radius $r \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5\}$ and $k = 3$. Here, the radius signifies the ratio between the number of clauses m and the number of variables n.

4.2 Hyperparameters Searches

Our ablation study encompassed three distinct rounds of hyperparameter searches for each instance. The inaugural search assumption was concentrated on the exploration of various decoders—LSTM, GRU, and Transformer—along with adjusting associated hyperparameters. However, during this phase, we did not contemplate baselines beyond the zero value nor did we consider variables' representation techniques beyond One-Hot encoding.

To compare against the first round, the subsequent round introduced an additional degree of flexibility by integrating all the proposed baselines. This phase also presented the option of eschewing the utilization of any baseline, i.e., making the baseline equivalent to zero. Despite this change, the variables' embedding technique remained unaltered as One-Hot encoding.

In the third assumption, the exploration expanded to include Node2Vec embeddings, which are adept at capturing intricate relationships among variables and clauses. This provided a platform to ascertain the influence of variables' representation on the model's performance.

It's noteworthy that this investigation also encapsulated all previously delved into elements, including decoder architectures, baselines, and One-Hot encoding embeddings. As mentioned before, each of these rounds integrated additional

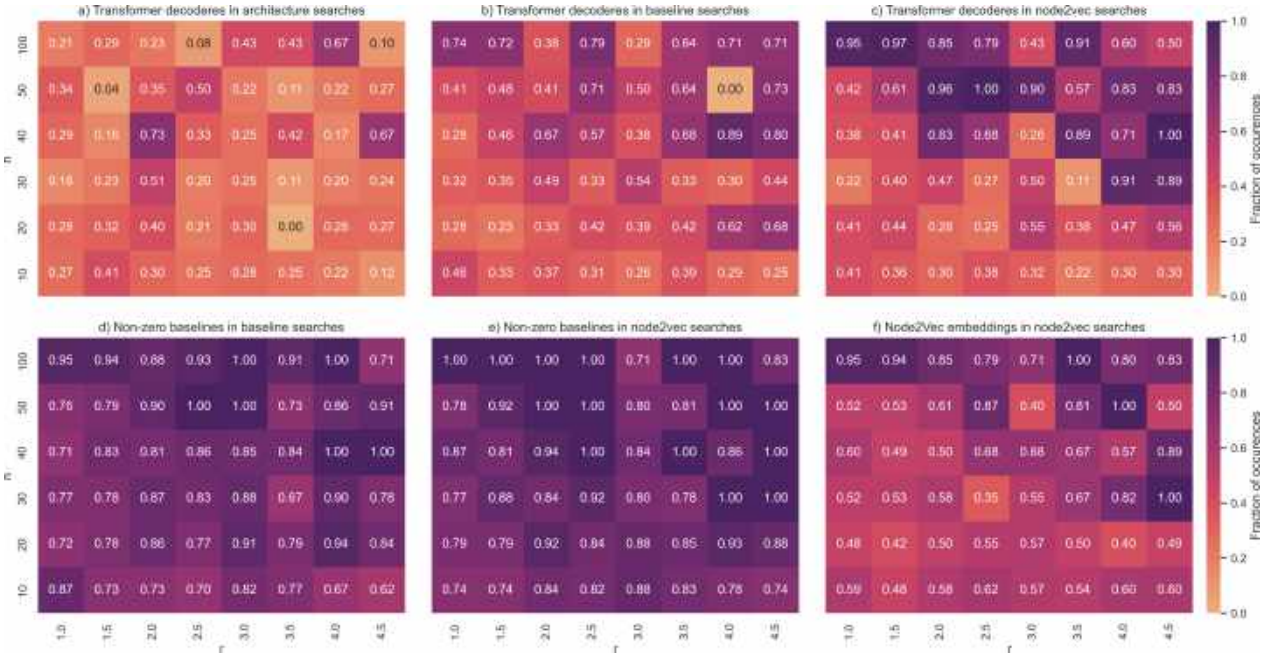


Fig. 8. Fraction of times Transformer decoders, non-zero baselines, and Node2Vec embeddings are present in the best trials for different values of n, r, and search assumptions. Sub-figures *a)*, *b)*, and *c)*, correspond to the fraction of times a Transformer decoder is present in the best trials when the search assumption was architecture, baseline, and Node2Vec, respectively. Sub-figures *d)* and *e)* show the fraction of times a non-zero baseline is present in the best trials when the search was baseline and Node2Vec, Sub-figure *f)* presents the fraction of times a Node2Vec embedding was present in the best trials in the Node2Vec search

hyperparameters into the searches, such as the learning rate, the number of layers, embedding size, logit clipping, logit temperature, among others.

For the systematic execution of each hyperparameter search, we leveraged the Tune [21] and Optuna [1] frameworks. The Tree-structured Parzen Estimator (TPE) [7] was employed as our hyperparameter search algorithm, complemented by the Asynchronous HyperBand Scheduler (ASHA) [20] to preemptively terminate unpromising trials.

The configuration for the ASHA scheduler was set with a grace period of $((2 \cdot n) + m) \times 4$ samples, and a cap at $((2 \cdot n) + m) \times 64$ samples. The TPE was designed to conduct 50 trials, with each batch constituted by 32 replicas of the identical instance.

The primary objective was to pinpoint the optimal configuration that maximized the satisfied clauses during the model's evaluation

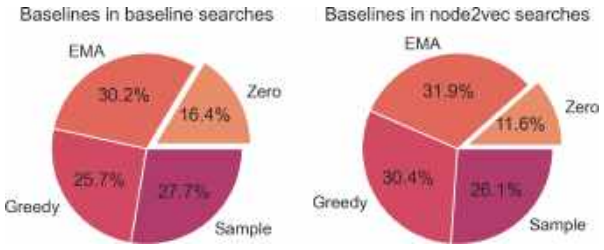


Fig. 9. Average fraction of times the different baselines are present in the best trials

phase. Pertaining to the hyperparameter search associated with the Node2Vec representations, we deployed the TPE for a total of 35 trials.

The ASHA scheduler in this context operated with a grace period of 5, peaking at 25 epochs. To guarantee a robust representation of the variables, the embedding size was set at 128.

4.3 Evaluation

To evaluate the performance of a specific model configuration on a given instance (i.e., a trial within a hyperparameter search), we adopted the following procedure: After processing every 320 samples, which comprise 10 batches with each batch having a size of 32 during the optimization process, we sampled 128 potential assignments (that is, we ran an episode on a single batch of size 128).

Subsequently, we computed the number of clauses satisfied (sat) by each of these assignments. The count of sat clauses at this point is determined by the highest number of sat clauses among the 128 assignments. The performance metric for the model, represented by the number of sat clauses achieved by a particular configuration, is the maximum number observed throughout the entire trial.

Furthermore, the number of sat clauses attained during a hyperparameter search on a given instance is derived from the maximum number of sat clauses observed across all 50 trials. Any trial that matches this maximum value is deemed as a best trial.

5 Results

After obtaining the number of sat clauses and identifying the best trials from the hyperparameter searches conducted for each combination of n, r, and search assumption (architecture, baseline, and Node2Vec embedding), we analyzed the impact of these search assumptions on the number of clauses the model satisfies.

Additionally, we delved into the architectures, baselines, and variables' representations involved in the top trials as both the number of variables and radius increased. This provided valuable insights into the configurations that yield superior solutions in terms of satisfied clauses.

5.1 Does the Search Assumption Impact the MaxSAT Solutions?

We used line graphs to illustrate the fraction of satisfied clauses achieved by each search assumption as the radius r increased. Moreover, a 99%-ile confidence interval was included to highlight the uncertainty in the results. Figure 5 shows the average fraction of sat clauses over different n values.

The graph reveals that, on average, integrating baselines into the search led to an increase in the number of satisfied clauses. This trend persisted across various r values. Additionally, incorporating Node2Vec representations into the search process yielded further enhancements in the number of satisfied clauses. This evidence suggests that the inclusion of baselines and Node2Vec embeddings can markedly improve the performance of the search algorithm.

To offer a more granular understanding of the influence of different search assumptions on the optimization process, we have included additional graphs for each n value. Figure 6 illustrates the line graphs for $n = 10$, 20, 30, 40, 50, and 100.

Examining these charts, we discern an interesting observation for smaller n values, such as 10, 20, and 30. For these cases, the benefit of integrating baselines or Node2Vec into the search space is less discernible.

For instance, with $n = 10$, all search methodologies produce similar outcomes, suggesting our optimization procedure is adept at identifying robust solutions irrespective of search assumptions.

When scrutinizing the outcomes for $n = 20$ and 30, we notice that the methods using Node2Vec do not consistently excel beyond other assumptions. For $n = 20$, the most optimal outcomes, on average, emerge when only baselines are integrated into the search space.

Likewise, for $n = 30$, no singular assumption consistently prevails across varying radii. This insinuates that, with smaller instances and an insufficient number of trials in the hyperparameters search, a more streamlined search space might occasionally offer superior outcomes. Yet, we posit that augmenting the number of trials in the

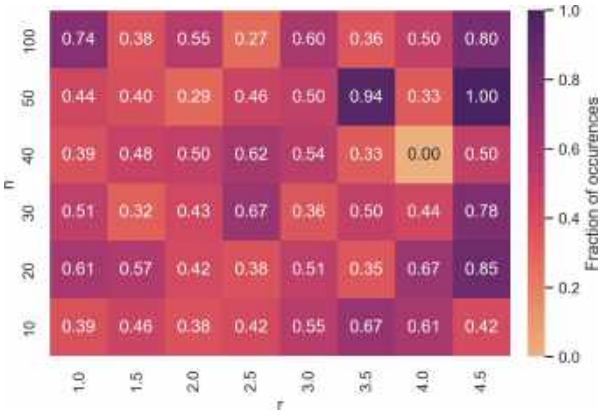


Fig. 10. Fraction of times the context was used in the best trials for the different values of n and r in the Node2Vec searches. Results are normalized by the fraction of times the Node2Vec representations were chosen

hyperparameters search could bolster results for both the baseline and Node2Vec assumptions.

Conversely, for greater n values, namely 40, 50, and 100, the merit of embedding baselines in the search space becomes evident. These visuals clearly depict an augmented fraction of satisfied clauses when baselines are incorporated, attesting to the efficacy of this approach for more extensive problem instances.

Moreover, the integration of Node2Vec embeddings into the search space results in general in even more pronounced improvements in satisfied clauses, emphasizing the advantages of incorporating instance-specific information into the optimization process. In light of the aforementioned analyses and observations, it becomes evident that search assumptions significantly influence MaxSAT solutions. These revelations accentuate the importance of employing baselines and leveraging Node2Vec embeddings, particularly when addressing problems of larger scales.

5.2 Which Configuration is Better?

To determine the most effective configurations, we counted the number of best trials as both n and r increased. Figure 7 demonstrates that for simpler instances—those with smaller n and r values—many trials achieve the sat number of clauses (i.e., the maximum number of satisfied clauses achieved by the model during a specific search).

This indicates consistent model performance even with randomized hyperparameters during the hyperparameters searches' warm-up phase. However, for more complex instances (larger n or r values), reaching the maximum number of satisfied clauses is challenging, and only specific configurations yield the best results.

Figure 8 provides insights into how frequently Transformer decoders, non-zero baselines, and Node2Vec representations appear in the best trials across the values of n, r, and search assumptions. For different search assumptions, sub-figures 8(a-c) show the prevalence of Transformer decoders.

With increasing n and r, the Transformer decoder becomes more favored over LSTM and GRU architectures, especially when the search includes baselines or variables' representation based on Node2Vec. Sub-figures 8(d-e) reveal a consistent preference for non-zero baselines across varying n and r values when available in the search space.

This suggests their pivotal role in optimizing MaxSAT solutions. Sub-figure 8(f) illustrates the preference for Node2Vec representations during Node2Vec searches. When available, these embeddings are often chosen, signaling the importance of instance-specific information, particularly for more intricate instances.

While simpler instances often exhibit a range of top-performing configurations, as revealed in fig. 7, complex ones with larger n or r values are more reliant on specific elements like Transformer decoders, non-zero baselines, and Node2Vec embeddings. This underscores the significance of these elements for handling intricate instances. Figure 9 offers a breakdown of how often different baselines feature in top trials.

It becomes clear that non-zero baselines play a substantial role in achieving optimal solutions. The pie chart highlights the dominance of the EMA approach in both searches, with greedy and sample baselines also being prominent. These data emphasize the merit of considering diverse baseline strategies in optimization processes.

During our experiments, models had the option to either utilize or ignore the context when Node2Vec variables were selected. Figure 10 portrays how often context was incorporated in top trials, normalized for instances when Node2Vec-type variables were selected. While its adoption slightly increases with n and r, context is not consistently present in top trials. We theorize that refining the context design might bolster the information from Node2Vec variables, potentially enhancing the solutions.

6 Conclusion and Future Work

We conducted a series of experiments across a diverse set of random 3-SAT instances to assess the potential of a sequential model for approximating solutions to the MaxSAT problem optimizing the parameters of the parametric model by maximizing the expected value of the number of satisfied clauses to predict a distribution over the possible solutions using the policy gradient method.

Our goal was to discern the architectural and procedural elements that enhance the optimization process to maximize the number of satisfied clauses.

The experiments presented demonstrated the potential advantages of using certain configurations and search assumptions, such as the inclusion of baselines and the Node2Vec embeddings, in improving the efficiency and accuracy of the model for MaxSAT solutions.

The clear influence of Transformer decoders, particularly for more complex instances, emphasized the utility of architectures that can capture long-range dependencies in the problem. This is pivotal for tasks such as MaxSAT where inter-variable relationships can have a significant impact on the overall satisfaction of clauses.

The introduction and evident advantage of non-zero baselines, especially the EMA approach, signaled the importance of stable and guided optimization. In the same way, Node2Vec embeddings benefit the model, especially as n and r increase. However, one surprising outcome was the inconsistent utilization of context with Node2Vec variables in top trials.

We postulate that an improved context design, which is a promising avenue for improvement, could supplement rich semantic information, potentially leading to enhanced solutions. Moving forward, several research directions emerge from the insights and limitations of our current approach:

- **Advanced Variables Representations**. Instead of relying solely on Node2Vec, we propose exploring more sophisticated graph embedding techniques, especially Graph Neural Networks, to better capture the structural intricacies of the MaxSAT instance.
- **Refining Context Usage**. The ambiguous outcomes from our usage of context with Node2Vec variables highlight the need for further investigation. A more refined integration approach, such as a time-dependent context updated at each time step using attention mechanisms, is worthy of exploration.

  Coupling this with alternative graph embedding methods may offer richer structural insights into the SAT formula.
- **Alternative Architectural Decoder Considerations**. The Transformer model's efficacy, especially for complex instances, leads us to believe there's potential in transformer-tailored architectures for combinatorial optimization tasks.

  For instance, a specialized decoder design that isn't reliant on consuming all previous inputs directly, and instead uses context more effectively, might yield better results.
- **Expanded Dataset Variability**. While our current experiments leaned on random 3-SAT instances, there's merit in diversifying our test cases. This includes exploring other MaxSAT problem types and integrating real-world instances to gain a comprehensive understanding of the model's capabilities.
- **Enhanced Exploration Techniques**. The logit temperature and clipping methods served us well in this study. Yet, integrating other exploration or regularization techniques, like the entropy bonus in computing the policy

gradient loss, might elevate the quality of the models outputs.

– **Improve Post-Processing**. Post-model solution refinement is a promising direction. Leveraging larger samplings, advanced techniques like beam search, or tailored search strategies could maximize solution quality further.

– **Contemplation of Variable Ordering**. Throughout our experiments, the variables were introduced to the model in ascending order based on their index. Investigating other types of variable ordering, especially those based on advanced strategies, may prove advantageous in the search for optimal assignments.

Acknowledgments

Authors would like to acknowledge the support provided by the Instituto Politécnico Nacional under projects: 20200651, 20210316, 20220002, 20230232, 20220798 and 20211096 to carry out this research. O. Gutiérrez thanks CONAHCYT for the scholarship granted towards pursuing his graduate studies.

References

1. **Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M. (2019).** Optuna: A next-generation hyperparameter optimization framework. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2623–2631. DOI: 10.1145/3292500.3330701.
2. **Avellaneda, F. (2020).** A short description of the solver EvalMaxSAT. MaxSAT Evaluation, Vol. 8.
3. **Bacchus, F. (2022).** MaxHS in the 2022 MaxSAT evaluation. MaxSAT Evaluation 2022, Vol. B-2022, pp. 17–18.
4. **Bello, I., Pham, H., Le, Q. V., Norouzi, M., Bengio, S. (2016).** Neural combinatorial optimization with reinforcement learning. 5th International Conference on Learning Representations. DOI: 10.48550/arXiv.1611.09940.
5. **Bengio, Y., Lodi, A., Prouvost, A. (2021).** Machine learning for combinatorial optimization: A methodological tour d'horizon. European Journal of Operational Research, Vol. 290, No. 2, pp. 405–421. DOI: 10.1016/j.ejor.2020.07.063.
6. **Berg, J., Demirović, E., Stuckey, P. J. (2019).** Core-boosted linear search for incomplete MaxSAT. In: Rousseau, L. M., Stergiou, K. (eds), Integration of Constraint Programming, Artificial Intelligence, and Operations Research. CPAIOR 2019. Lecture Notes in Computer Science, Vol. 11494, pp. 39–56. DOI: 10.1007/978-3-030-19212-9_3.
7. **Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B. (2011).** Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems, Vol. 24.
8. **Cho, K., van-Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014).** Learning phrase representations using RNN encoder-decoder for statistical machine translation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). DOI: 10.48550/arXiv.1406.1078.
9. **Davies, J. (2013).** Solving MaxSAT by decoupling optimization and satisfaction. Ph.D. thesis, University of Toronto.
10. **Demirovic, E., Musliu, N. (2014).** Modeling high school timetabling as partial weighted maxSAT. LaSh 2014: The 4th Workshop on Logic and Search, pp. 1–39.
11. **Demirovic, E., Musliu, N., Winter, F. (2019).** Modeling and solving staff scheduling with partial weighted maxSAT. Annals of Operations Research, Vol. 275, pp. 79–99. DOI: 10.1007/s10479-017-2693-y.
12. **Graves, A. (2013).** Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850. DOI: 10.48550/arXiv.1308.0850.

13. **Grover, A., Leskovec, J. (2016).** node2vec: Scalable feature learning for networks. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge discovery and data mining, pp. 855–864. DOI: 10.1145/2939672.2939754.
14. **Hochreiter, S., Schmidhuber, J. (1997).** Long short-term memory. Neural computation, Vol. 9, No. 8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
15. **Hottung, A., Kwon, Y. D., Tierney, K. (2021).** Efficient active search for combinatorial optimization problems. 2022 The International Conference on Learning Representations. DOI: 10.48550/arXiv.2106.05126.
16. **Juma, F., Hsu, E. I., McIlraith, S. A. (2012).** Preference-based planning via MaxSAT. Advances in Artificial Intelligence: 25th Canadian Conference on Artificial Intelligence, Vol. 7310, pp. 109–120. DOI: 10.1007/978-3-642-30353-1_10.
17. **Khalil, E., Dai, H., Zhang, Y., Dilkina, B., Song, L. (2017).** Learning combinatorial optimization algorithms over graphs. Advances in neural information processing systems, Vol. 30.
18. **Kingma, D. P., Ba, J. (2014).** Adam: A method for stochastic optimization. International Conference on Learning Representations. DOI: 10.48550/arXiv.1412.6980.
19. **Kool, W., van-Hoof, H., Welling, M. (2018).** Attention, learn to solve routing problems! 2019 International Conference on Learning Representations. DOI: 10.48550/arXiv.1803.08475.
20. **Li, L., Jamieson, K., Rostamizadeh, A., Gonina, K., Hardt, M., Recht, B., Talwalkar, A. (2018).** Massively parallel hyperparameter tuning. ICLR 2018 Conference Acceptance Decision.
21. **Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., Stoica, I. (2018).** Tune: A research platform for distributed model selection and training. arXiv. DOI: 10.48550/arXiv.1807.05118.
22. **Martins, R., Manquinho, V., Lynce, I. (2014).** Open-WBO: A modular MaxSAT solver. Theory and Applications of Satisfiability Testing–SAT 2014: 17th International Conference, Held as Part of the Vienna Summer of Logic, VSL 2014, pp. 438–445. DOI: 10.1007/978-3-319-09284-3_33.
23. **Martins, R., Manthey, N., Terra-Neves, M., Manquinho, V., Lynce, I. (2023).** Open-WBO @ MaxSAT evaluation 2023. MaxSAT Evaluation 2023, pp. 18–19.
24. **Safarpour, S., Mangassarian, H., Veneris, A., Liffiton, M. H., Sakallah, K. A. (2007).** Improved design debugging using maximum satisfiability. Formal Methods in Computer Aided Design, pp. 13–19. DOI: 10.1109/FAMCAD.2007.26.
25. **Selsam, D., Lamm, M., Bünz, B., Liang, P., de-Moura, L., Dill, D. L. (2018).** Learning a SAT solver from single-bit supervision. arXiv. DOI: 10.48550/arXiv.1802.03685.
26. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2017).** Attention is all you need. Advances in Neural Information Processing Systems, Vol. 30.
27. **Williams, R. J. (1992).** Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, Vol. 8, pp. 229–256. DOI: 10.1007/BF00992696.

Article received on 26/10/2023; accepted on 24/11/2023.
* *Corresponding author is Ricardo Menchaca-Méndez.*

ISSN 2007-9737

Impact of 5G Technology on Cybersecurity: A Comprehensive Systematic and Bibliometric Review

Javier Gamboa-Cruzado[1,*], Luis Cuya-Chuica[2], Jefferson López-Goycochea[3], Angel Nuñez-Meza[4], Carlos Del-Valle-Jurado[5]

[1] Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Peru

[2] Universidad Nacional Federico Villareal, Facultad de Ingeniería Industrial y de Sistemas, Peru

[3] Universidad de San Martín de Porres, Facultad de Ingeniería y Arquitectura, Peru

[4] Universidad Nacional Daniel Alcides Carrión, Facultad de Ingeniería de Sistemas, Peru

[5] Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería Geológica, Minera, Metalúrgica y Geográfica, Peru

jgamboa65@hotmail.com, 2018006116@unfv.edu.pe, jlopezg@usmp.pe, anunezm@undac.edu.pe, cdelvallej@unmsm.edu.pe

Abstract. The integration of 5G technology is progressively becoming standardized in everyday life, presenting notable benefits. However, it also remains largely uncharted territory, whether due to the diversity of applicable areas or its innovative nature. As such, there is a growing interest in delving into its various fields of application. In this context, the aim of the present research is to discern the state of the art of 5G technology and its impact on cybersecurity. To achieve this goal, a systematic review of studies published between 2016 and 2022 was conducted. The search strategy employed yielded a total of 13,235 papers from recognized sources such as Scopus, Web of Science, ARDI, ACM Digital Library, IEEE Xplore, and EBSCOhost. From this set, 68 papers were identified as relevant studies, after applying the established filters and exclusion criteria. Among the derived findings, there is a notable concentration of bibliometric flow by countries, the various areas of application, and the co-occurrence of organizations that previously researched this topic. The main implication of this research lies in identifying a tangible need to increase and improve studies regarding the application of 5G technology and its impact on cybersecurity.

Keywords. Technology, 5G, cybersecurity, systematic review, bibliometric review.

1 Introduction

At this stage of technological progress, as 5G technology has become an integral part of our daily lives, its application is evident in various areas such as: communication between devices on the mobile network [81], data transmission [82], and network slicing [83].

This fifth generation, by offering a broader array of services compared to its predecessors, allows for a wider application in areas like the interconnection of open systems [84]. However, concerns arise regarding security and privacy [85], given the emerging vulnerabilities.

Despite these advancements, a comprehensive understanding of the state of the art concerning this technology's impact on cybersecurity is still lacking, which, if explored, could pave the way for future developments in this domain. Recent studies indicate that the 5G generation is designed to offer faster speeds, lower latency, and a more robust connection compared to previous communication technologies.

However, a heightened security threat is anticipated due to the wide range of vectors through which adversaries can launch attacks. Consequently, cybersecurity becomes crucial, as it is associated with the protection of confidential data and user personal information [78], key elements to counteract such threats.

On the other hand, the impacts of 5G technology on cybersecurity [76] are promising, considering that any traditional system involves restricting cyber access to confidential data and components, which is expected to benefit the user. It is suggested to conduct a holistic assessment to understand the attack surface of a 5G-based system, with the aim of comprehending the potential cyber risk for such infrastructure and developing appropriate mechanisms for protection against these threats [73].

Furthermore, it is necessary to delve into the different definitions addressed in the review and conduct a classification of the various forms of existing information [75], to gain a clearer and structured understanding of the cybersecurity landscape in the 5G environment. It is evident that 5G technology applications can enhance cybersecurity and provide an overall better service; however, there is not yet specific research that analyzes the impact of this technology on cybersecurity.

This paper aims to determine the state of the art of 5G technology and its impact on cybersecurity. The structure of the document is organized as follows: Section II presents the theoretical framework; Section III describes the review method used; Section IV highlights the results and derived discussions; and finally, Section V provides the conclusions and suggests directions for future research in this field.

2 Background and Related Work

2.1 Research Problems and Objectives

To adequately understand 5G technology, which is already widely used today, it's essential to review some fundamental concepts.

2.1.1 First Generation (1G)

Mobile systems have evolved significantly since the emergence of the first generation in the 1980s. Back then, there were no robust security mechanisms [36].

2.1.2 Second Generation (2G)

The second generation of mobile phone services, known as 2G, was introduced in 1991, succeeding 1G. Mobile devices equipped with this technology offered voice and messaging services. However, 2G posed a series of security challenges. Within the realm of 2G networks, cyber attackers often used spam tactics as a means to distribute unsolicited information to users. This strategy facilitated the spread of malicious code among mobile device users. Attackers were using malicious code for harmful purposes [53].

2.1.3 Third Generation (3G)

The security system in 3G considered all the vulnerabilities detected in 2G and proceeded to rectify them. The security architecture implemented in 3G communications was articulated into five components: network access security, network domain security, user domain security, application security, and visibility and configurability security. Some of these vulnerabilities were related to unauthorized acquisition of users' confidential information, illicit interventions, and identity spoofing attacks [84].

2.1.4 Fourth Generation (4G)

The 4G security architecture was developed based on experiences and lessons drawn from the 2G and 3G networks. 4G introduced a revamped set of cryptographic algorithms and a significantly

different key structure compared to 2G and 3G. However, 4G also inherited certain security problems present in these earlier networks. With the adoption of 4G technology, mobile operators had the capability to offer novel services, including those at high speeds. It is important to note that cybercriminals demonstrated an organization and adaptability that surpassed expectations [85].

2.1.5 Fifth Generation (5G)

Currently, a vast amount of information is in multimedia formats, such as images and videos. Given the increasing demand for this data, multimedia exchange has positioned itself as one of the most sought-after Internet services. Concurrently, the emergence of fifth generation (5G) networks brings numerous benefits in terms of providing these multimedia sharing services.

It is imperative to note that, given the relevance of image exchange in the multimedia context [47], the 5G network demands the implementation of security mechanisms adapted to the new applications, network architectures, and air interface technologies [77].

2.1.6 Future of the Network (6G)

The 6G promises to incorporate various components, such as edge computing, cloud computing, and artificial intelligence. Among these elements, the communications infrastructure expects to have the largest market share, reaching figures of up to one billion US dollars [29].

2.2 Cybersecurity

Cybersecurity is recognized as an essential element in the implementation of 5G technology within the European Union (EU). It is projected that 5G networks will play a leading role in the Digital Single Market (DSM), significantly influencing areas such as energy, transportation, and health services.

Furthermore, with the emergence of 5G, we are heading towards an even more interconnected world. In this context, vulnerabilities detected in the 5G systems of a member state could impact the entire EU. Consequently, it is essential to promote collaboration and cooperation among nations to ensure a safe and coordinated implementation of 5G [50].

3 Review Method

To conduct the systematic review on the impact of 5G technology on cybersecurity, the principle proposed by Petersen [69] and Rimaki [70] was followed.

They suggest the formulation of research questions, the search in relevant sources to extract data, and answering these questions considering the identified limitations and conclusions.

For more details, refer to Figure 1. A keyword map was also used as suggested by Kitchenham [71], with the aim of showing the most predominant research trends and detecting the topics present within the analyzed field.

A word map facilitates visualization and a better understanding of the actual meaning of each concept developed on the topic, as explained by Linnenluecke [72].

3.1 Research Problems and Objectives

A general research question (RQG) is formulated, and additionally, several specific research questions (RQ) are proposed along with their respective objectives, as shown in Table 1 below.

3.2 Information Sources and Search Strategies

The sources selected for this systematic review include IEEE Xplore, ARDI, Web of Science, EBSCOhost, Scopus, and ACM Digital Library. To conduct the information search related to both dependent and independent variables, synonyms were sought, considering how authors refer to them in their papers, as shown in Table 2.

To track research papers based on search descriptors, equations were initially used in the selected sources. Boolean logic was also applied, tailoring it to the specifics of each source, as shown in Table 3.

3.3 Identified Studies

In this section, the number of results obtained per source for the research is presented, as shown in Figure 2.

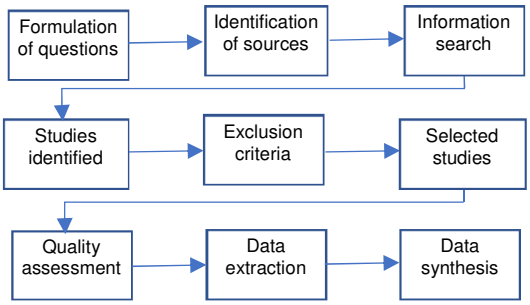


Fig. 1. RSL process

3.4 Exclusion Criteria

After obtaining the papers, the next step is to select them for evaluation to determine if they are suitable for the research. For this purpose, four filters with eight exclusion criteria (EC) were used, detailed as follows:

EC1: The papers are more than 7 years old.

EC2: The papers are not written in English.

EC3: The papers have not been published in conferences or journals.

EC4: The papers are systematic reviews.

EC5: The titles and keywords of the papers are not descriptive or relevant.

EC6: Full text of the papers is not available.

EC7: The papers are not unique.

EC8: The paper is less than 10 pages long.

3.5 Study Selection

Next, the selection of studies is presented using the PRISMA diagram, as shown in Figure 3.

3.6 Quality Assessment

At this stage, the quality of the 68 selected papers was assessed after applying the exclusion criteria. To ensure the quality of the papers, 7 quality criteria (QA) were established for their evaluation, which are as follows:

QA1: Does the paper consider fundamental research?

QA2: Does the paper reference the instruments used for data collection?

QA3: Does the paper provide access to the full text of the research?

QA4: Does the paper present a clear and precise delimitation of the specific area it addresses in its research?

QA5: Does the paper provide a comprehensive explanation of the context in which the research was conducted?

QA6: Does the researcher have relevant academic training in the field of study of 5G Technology and Cybersecurity?

QA7: Does the researcher provide contact information or institutional affiliation for future inquiries?

The assessment of the QA questions is conducted using a rating scale that ranges from 1 to 3 (1- Not good, 2- Good, and 3- Very good). The minimum value for inclusion is 12 (60% of the maximum). 68 papers were chosen as they scored ≥ 12. Table 4 displays the results of the quality assessment.

3.7 Data Extraction Strategies

Now, the process began to extract the most relevant papers with the aim to precisely address the established research questions. The extracted information included: Reference number, paper title, URL, source, year, countries, ISSN, type of publication, name of the publication, authors, affiliations, quartile, H-Index, research methodology, number of citations, abstract, keywords, discussion, and conclusion. To categorize the papers, Mendeley Desktop tool was utilized, as shown in Figure 4.

3.8 Synthesis of Findings

The information extracted for the research questions (RQ) was tabulated and presented as quantitative data, which was used to conduct a statistical comparison between the different findings corresponding to each research question.

These data obtained facilitated the identification of certain research patterns that have manifested over the past seven years.

4 Results and Discussion

In this section, key findings derived from the systematic review of the analyzed papers are presented and discussed. We examine the most prominent application areas, the quality levels of the sources, the co-authorship networks of the most influential researchers, the geographical distribution of the research, and the clusters of papers with similarity in titles.

This section provides a comprehensive view of the results obtained, contributing to a deeper understanding of this field of study and its evolution in recent years.

4.1 General Description of the Studies

The general findings obtained in this review are presented below. Figure 5 illustrates the percentage of papers according to the types of publication. In this study, 68 papers were selected, setting the criteria to only include papers from journals and conferences. Figure 5 reveals that 97.1% are from journals, while only 2.9% are from conferences, out of the total types of papers considered.

In the research conducted by Farooqui, Arshad, and Khan [73], both types of papers—conferences and journals—were also considered, aligning with the criteria used in this paper. On the other hand, Raveendran and Tabet [74] also used 68 papers in their review.

Authors Lozano and Mateo [75] state that in their paper, they employed 62.42% from conferences, 30.96% from journals, and included books, which constituted 6.62% of the participation, thus differentiating their review in terms of the exclusion criteria of the type of paper used in the current research.

It is noted that, over the years, the authors of the reviews are considering approximately 70 papers for their research, in addition to prioritizing papers from journals and conferences in their exclusion criteria.

In the future, authors might consider papers from journals and conferences as primary information, given the preference for these types of publications in this research field, as reflected in the various reviews considered. Table 5 displays the number of papers by continent and year.

Table 1. Research questions and objectives

Research Question	Objectives
RQG: What is the state of the art in research regarding 5G Technology and its impact on Cybersecurity?	Determine the state of the art in research regarding 5G Technology and its impact on Cybersecurity.
RQ1: In which industrial sectors or fields of study is 5G Technology having a significant impact?	Determine the industrial sectors or fields of study where 5G Technology is influential.
RQ2: How are the publications on 5G Technology and its impact on Cybersecurity distributed among the different journal quartiles?	Identify the distribution of publications on 5G Technology and its impact on Cybersecurity among different journal quartiles.
RQ3: Who are the most prolific or influential authors in the field of 5G Technology and its impact on Cybersecurity, and what are their co-authorship networks?	Determine authors who frequently co-author papers on 5G Technology and its impact on Cybersecurity.
RQ4: Which countries lead in producing research on 5G Technology and its impact on Cybersecurity, and how are the bibliometric flows distributed among these countries?	Identify countries that often feature bibliometric flows in research on 5G Technology and its impact on Cybersecurity.
RQ5: How can papers be grouped based on the thematic similarity of their titles in the field of 5G Technology and its impact on Cybersecurity, and which topics dominate each cluster?	Identify clusters of papers whose titles show similarity in research about 5G Technology and its impact on Cybersecurity.

Table 2. Search descriptors and their synonyms

Descriptor	Description
5g technology/ 5g/ 5g network/ 5g mobile telephony/ fifth generation	Independent Variable
cybersecurity / online safety / security/cyber	Dependent Variable

Table 3. Information sources and search equation

Source	Search equation
Scopus	TITLE-ABS-KEY (("5g technology" OR 5g OR "5g network" OR "5g mobile telephony" OR "fifth generation") AND (cybersecurity OR "online safety" OR security OR cyber))
Web of Science	("5g technology" OR 5g OR "5g network" OR "5g mobile telephony" OR "fifth generation") AND (cybersecurity OR "online safety" OR security OR cyber) (Title) OR ("5g technology" OR 5g OR "5g network" OR "5g mobile telephony" OR "fifth generation") AND (cybersecurity OR "online safety" OR security OR cyber) (Abstract)
IEEE Xplore	(("Document Title":5g OR "Document Title":"5g technology" OR "Document Title":"5g network" OR "Document Title":"5g mobile telephony" OR "Document Title":"fifth generation") AND ("Document Title":cybersecurity OR "Document Title":"online safety" OR "Document Title":security OR "Document Title":cyber)) OR (("Abstract":5g OR "Abstract":"5g technology" OR "Abstract":"5g network" OR "Abstract":"5g mobile telephony" OR "Abstract":"fifth generation") AND ("Abstract":cybersecurity OR "Abstract":"online safety" OR "Abstract":security OR "Abstract":cyber))
ARDI	((Abstract:(\("5g technology" OR 5g OR "5g network" OR "5g mobile telephony" OR "fifth generation"\) AND \(cybersecurity OR "online safety" OR "security" OR "cyber"\))) OR (TitleCombined:(\("5g technology" OR 5g OR "5g network" OR "5g mobile telephony" OR "fifth generation"\) AND \(cybersecurity OR "online safety" OR "security" OR "cyber"\))))
EBSCO host	AND ("5g technology" OR 5g OR "5g network") AND (cybersecurity OR "online safety" OR security) Title OR ("5g technology" OR 5g OR "5g network") AND (cybersecurity OR "online safety" OR security) Abstract
ACM Digital Library	[[[Title: "5g technology"] OR [Title: 5g] OR [Title: "5g network"] OR [Title: "5g mobile telephony"] OR [Title: "fifth generation"]] AND [[Title: cybersecurity] OR [Title: "online safety"] OR [Title: security] OR [Title: cyber]]] OR [[[Abstract: "5g technology"] OR [Abstract: 5g] OR [Abstract: "5g network"] OR [Abstract: "5g mobile telephony"] OR [Abstract: "fifth generation"]] AND [[Abstract: cybersecurity] OR [Abstract: "online safety"] OR [Abstract: security] OR [Abstract: cyber]]]

The papers selected for this research cover the period from 2016 to 2022, with contributions from all continents. As shown, the year 2022 recorded the most contributions, while 2016 had the fewest contributions. In the study by Gamboa-Cruzado [89], a continuous increase in publications from 2016 to 2021 is observed, which coincides with the current research.

Figure 7 illustrates the number of papers contributed by each continent, classifying them into categories: good, average, and low. On the other hand, the previous figure reveals that the continents of Asia and Europe are classified as good, contributing more than 30 papers each, while Africa and Oceania are considered low, contributing fewer than 10 papers each.

In the study by Raveendran and Tabet [74], papers were considered over a broader time range, from 2000 to 2020, providing a more extended temporal window compared to the current paper. On the other hand, Lozano, and Mateo [75] align with the results presented in this paper in identifying Asia as the continent with the most publications, accounting for 39.65%, followed by Europe with 32.25%. In a study conducted by Zeb, Mahmood, Hassan, Piran, Guizani, and Gildlund [76], the chosen time range spanned from 2003 to 2021.

It can be concluded that, for potential future research on this topic, it would be prudent to prioritize the search for information on the Asian continent, given the trend observed in the results found, as well as in the research of other authors, suggests that this continent is the most productive in terms of contributions, followed by Europe.

Regarding temporality, it might be beneficial to broaden the range of years considered to gain a broader and possibly more representative view of the state of the art in the field of 5G technology and its impact on cybersecurity.

4.2 Answers to the Research Questions

Following are the conclusions and specific findings that address each of the research questions posed in the study.

Each research question is tackled individually, providing detailed analyses and answers based on the data gathered and assessed in the previous section. This part of the paper offers a clear structure for understanding how the results relate to the research objectives and provides valuable insights into the current state of the field in relation to the questions raised.

RQ1: In which industrial sectors or fields of study is 5G Technology having a significant impact?

The following presents Table 6, which illustrates the sectors where 5G technology finds greater application according to this review.

From the table, it is evident that the predominant areas of application for 5G technology in this study are physics at 58.8%, transport at 38.2%, and healthcare at 32.3%.

Raveendran and Tabet [74] identify main themes as categories in their research, leading them to formulate five master themes related to higher-order concerns, namely: electromagnetic pollution, cybersecurity issues, data center overload, proliferation of submarine cables, and electronic waste.

On the other hand, Zeb, Mahmood, Hassan, Piran, Guizani, and Gildlund [76] primarily focus on the information of the industrial digital twin for control and management processes in industrial applications. Still, they also touch upon cloud computing, machine learning, artificial intelligence, 5G industrial services, industrial twin placement strategies, green communication, and automated optical inspection.

Goudarzi, Ghayoor, Waseem, Fahad, and Traore [86] place importance on the realm of the Internet of Things in next-generation IoT-enabled smart grids, with a higher number of contributions derived from parts of their papers included in their review. Authors Sodhro, Awad, Beek, and Nikolakopoulos [87] examine network layer approaches and other technologies in their review, with the areas of network and security having the most contributions recorded.

Ferrag, Maglaras, Argyriou, Kosmanos, and Janicke [88] point out that the research area receiving the most contributions in the papers they selected is security and privacy. The diversity in the criteria mentioned by various authors when addressing 5G technology reflects the breadth and complexity of this emerging technology.

Based on the focus of the review, areas of physics and electromagnetism, as well as security, are particularly emphasized, indicating that these are significant areas of interest in the current 5G literature.

Physics and electromagnetism are fundamental to understanding and advancing 5G technology, as they lie at the core of how wireless signals are transmitted and received.

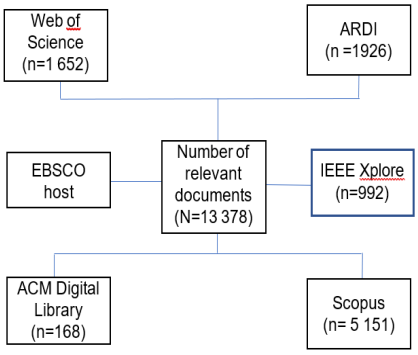


Fig. 2. Number of papers by source

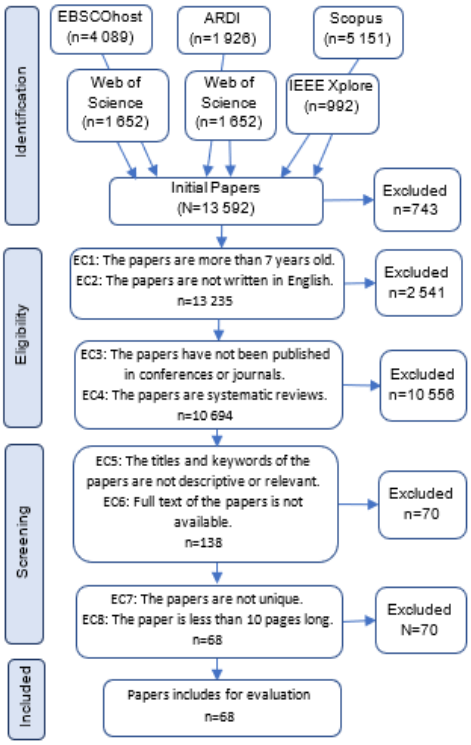


Fig. 3. PRISMA flow diagram

The implications of 5G technology in these fields are vast and could lead to new innovations or technical challenges that have not yet been fully explored. Future research in the field of 5G technology could greatly benefit from the findings and themes highlighted in the current review.

However, it is crucial for future researchers to also follow their own criteria and personal focus,

Table 4. Quality assessment of the papers

Ref.	QA1	QA2	QA3	QA4	QA5	QA6	QA7	Score
[1]	2	3	3	3	2	3	1	17
[2]	2	1	3	3	1	2	3	15
[3]	2	1	3	3	1	2	3	15
[4]	3	1	3	3	1	2	3	16
[5]	3	3	0	3	3	2	3	17
[6]	3	3	3	0	3	3	0	15
[7]	3	3	3	3	3	0	3	18
[8]	3	3	2	3	3	3	3	20
[9]	3	3	2	3	3	3	3	20
[10]	3	3	2	3	3	3	3	20
[11]	3	0	2	3	3	3	0	14
[12]	3	3	0	3	3	3	3	18
[13]	3	3	3	3	3	3	3	21
[14]	3	3	3	0	3	3	3	18
[15]	3	3	3	3	3	3	3	21
[16]	0	3	0	3	3	3	1	13
[17]	3	3	3	3	3	3	3	21
[18]	3	3	3	1	3	3	3	19
[19]	3	3	3	3	3	3	3	21
[20]	3	0	3	3	3	3	3	18
[21]	3	3	3	3	3	3	0	18
[22]	3	3	3	0	3	3	3	18
[23]	3	3	3	3	3	3	3	21
[24]	0	3	3	3	1	3	3	16
[25]	3	3	3	3	3	3	3	21
[26]	3	3	3	3	3	3	0	18
[27]	2	3	3	3	3	3	3	20
[28]	3	3	0	3	3	3	3	18
[29]	3	3	3	3	3	3	3	21
[30]	3	3	3	3	3	3	3	21
[31]	3	3	3	3	0	3	3	18
[32]	3	3	3	3	3	3	3	21
[33]	1	3	3	3	2	3	3	18
[34]	3	3	3	3	3	3	3	21
[35]	3	0	3	3	3	3	2	17
[36]	3	3	3	3	3	3	3	21
[37]	3	3	3	3	3	2	3	20
[38]	0	3	3	3	3	3	3	18
[39]	3	3	3	3	3	3	3	21
[40]	3	3	3	2	3	3	3	20
[41]	3	0	3	3	3	3	3	18
[42]	3	3	3	3	0	3	3	18
[43]	2	3	3	3	3	0	3	17
[44]	3	1	3	3	3	3	3	19
[45]	3	2	3	0	3	3	3	17
[46]	3	0	3	3	3	3	3	18
[47]	3	3	3	3	3	3	3	21
[48]	3	3	3	3	3	0	1	16
[49]	3	3	1	3	3	3	3	19
[50]	3	3	3	3	3	3	3	21
[51]	3	0	3	3	3	3	3	18
[52]	3	3	3	0	3	3	3	18
[53]	2	3	3	3	3	0	3	17
[54]	3	3	3	3	3	3	3	21
[55]	3	3	2	3	3	3	3	20
[56]	3	3	3	3	3	2	3	20
[57]	3	3	3	3	3	3	3	21
[58]	3	3	3	0	3	1	3	16
[59]	3	0	3	1	3	3	3	16
[60]	2	3	3	3	3	3	3	20
[61]	3	3	3	3	3	3	3	21
[62]	3	3	3	3	3	3	3	21
[63]	2	3	3	2	3	3	0	16
[64]	3	0	3	3	0	3	3	15
[65]	2	3	3	3	3	3	3	20
[66]	3	3	3	3	3	3	3	21
[67]	2	3	3	0	3	3	3	17
[68]	3	3	3	3	3	3	3	21

and perhaps explore other relevant areas or topics that might not have been extensively covered in current studies. The advice to adhere to the author's criteria suggests that each researcher should also bring their unique perspective and focus to the field, which could lead to new ideas and advancements not yet considered.

This is especially relevant in a rapidly moving and evolving technological field like 5G, where innovation and continuous exploration are essential for advancing knowledge and the practical application of the technology.

RQ2: What is the distribution of publications on 5G Technology and its impact on Cybersecurity among the different quartiles of journals?

Figure 6 displays the quartile levels broken down by year. The Figure 8 presented below

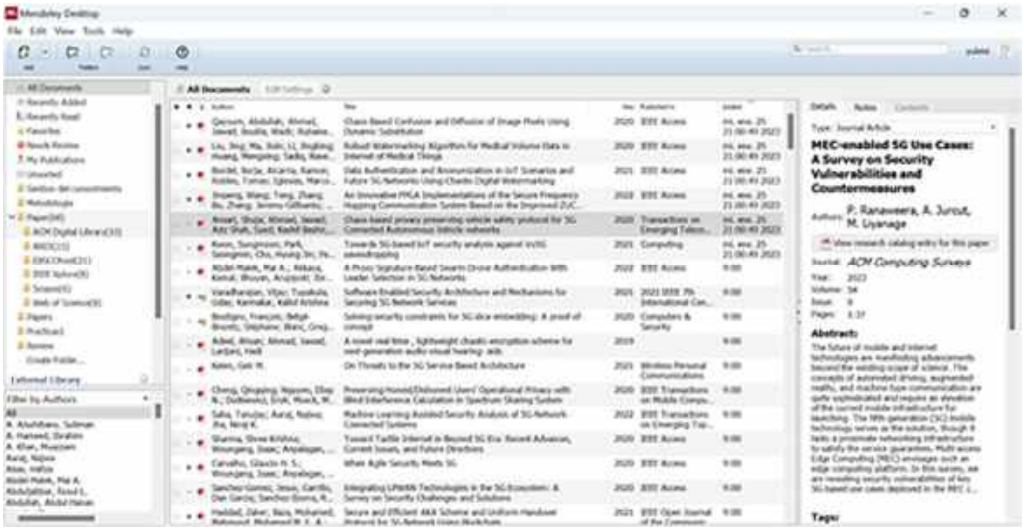


Fig. 4. Reporting of papers using Mendeley desktop

illustrates the relationship between the quartiles and the sources used in the review through a Sankey diagram. This type of diagram allows for a clear and effective visualization of how the different quartiles are distributed among the various reviewed sources, facilitating the understanding of the correlation between these two elements.

Through graphical flows, the Sankey diagram displays how the quartiles are associated with each source, providing an intuitive visual representation that can be greatly helpful in understanding the distribution and trend of the quartiles in relation to the sources consulted in the systematic review process.

The Figure underscores that papers in this research are categorized into quartile levels Q1, Q2, Q3, and Q4, with Q1 being the most prevalent. Moreover, it is observed that the year 2022 contributed the highest number of papers.

On the other hand, the correlation between quartiles and the sources from which the papers originated is illustrated, revealing that all papers categorized in the Q1 quartile come from all consulted sources, with the exception of EBSCOhost.

This analysis reflects not only the distribution of papers across different quartiles and their yearly evolution but also the relationship between the perceived quality of the papers, represented by the quartiles, and the sources from which they were drawn, providing an in-depth perspective on the provenance and quality of the papers reviewed in this research.

The authors Ogbodo, Abu-Mahfouz, and Kurien [77] also agree with the findings of this research, including the same quartiles in their paper, with the exception of Q2. This reflects a similarity in the distribution of the quality of papers across different studies, although with some differences regarding the Q2 quartile.

This might indicate variations in the selection or evaluation of papers among different studies. This overlap, with the mentioned exception, could suggest a trend or pattern in the distribution of papers based on their quality and provides a basis for comparing and contrasting results among different studies in the realm of 5G Technology and its impact on cybersecurity.

It is observed that the considered papers span all quartile levels, though there is a notable decrease in the Q2 category. This could be a point of consideration for future researchers looking to evaluate the quality of this type of research.

The variability in the quartile distribution, particularly the drop in Q2, may offer insights regarding the quality and relevance of existing studies in the realm of 5G Technology and its impact on cybersecurity.

Furthermore, this pattern could serve as a guide for future research, suggesting a more

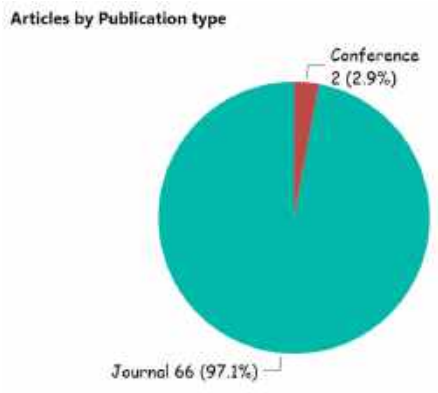


Fig. 5. Percentage of papers by journal and conference

Table 5. Number of papers by continent and year

Years / Continent	2016	2017	2018	2019	2020	2021
Asia	1	1	4	1	11	16
Europe	1	4	5	7	13	4
America	0	0	1	1	6	1
Oceania	0	0	0	0	2	1
Africa	0	0	0	0	1	
Total	2	5	10	9	33	22

thorough review of evaluation criteria and paper selection, especially concerning Q2.

This could contribute to a broader and more accurate understanding of the existing literature, and possibly influence the direction and focus of future research efforts in this field.

RQ3: Who are the most prolific or influential authors in the field of 5G Technology and its impact on Cybersecurity, and what are their co-authorship networks?

With respect to co-occurrence, there are semantic distributional models based on count vectors that represent by means of a matrix the frequency of occurrence of words (authors, in this case) in a document. The similarity between two words vectors can be obtained through the angle they form, specifically by the cosine of the angle.

It is considered that the smaller the angle, and consequently the cosine of the angle, the greater the similarity between them. With the following equation, we obtain the cosine measure between the documents d_i and d_j where d_{ik} is the weight of the semantic feature k in the document d_i:

$$\sin(d_i, d_j) = \cos(a) = \frac{\sum_{k=1}^{m}(d_{ik} * d_{jk})}{\sqrt{(\sum_{k=1}^{m} d_{ik}^2) * (\sum_{k=1}^{m} d_{jk}^2)}}. \quad (1)$$

Figure 9 illustrates the bibliometric network that highlights the frequency with which certain authors collaborate as co-authors in the studies considered in this review.

This visualization can provide a clear representation of the connections and collaborations among researchers within the study field of 5G Technology and its impact on cybersecurity. Through this network, it is possible to identify clusters of authors who frequently work together, which may indicate research groups or institutions that are actively contributing to this area.

Figure 9 displays the bibliometric network of co-authors in this review, highlighting four authors who show a recurring co-authorship frequency: Zhu Han, Latif U. Khan, Choong Seon Hong, and Ibrar Yacoob, who each contribute to three research papers respectively with other researchers.

This visualization can be significant in understanding existing collaborations and the network of professional relationships in the field of 5G Technology and its impact on cybersecurity. Identifying these recurring authors may point to individuals or research groups that have a notable influence or specialization in the area.

Lozano and Mateo [75] highlight two papers as the most cited in their research: "Performance evaluation of the IEEE 802.11p WAVE communication standard" and "Delay and broadcast reception rates of highway safety applications in vehicular ad hoc networks," both authored by Eichler.

These papers focus on the performance evaluation of vehicular communication standards and highway safety applications, respectively. On the other hand, Ly, and Yao [78] identify the publication "Deep learning in mobile and wireless networking: A survey" as the most recurring in their review, written by Zhang, Patras, and Haddadi.

This work encompasses a comprehensive review of deep learning applied to mobile and wireless networks, reflecting an interest in the

intersection between machine learning technologies and communication networks.

Based on the obtained results, it is evident that there are various prominent authors in the findings, which vary depending on the focus of the review. This suggests that in future research related to this topic, it might be beneficial to define one or several key benchmarks that could guide or inform the study in question.

Identifying these benchmarks can not only provide a solid foundation for the analysis but could also help situate the work within the broader context of the existing literature in this field. RQ4: What are the leading countries in the production of research on 5G Technology and its impact on Cybersecurity, and how are the bibliometric flows distributed among these countries?

The bibliometric flow between two countries can be calculated using the co-authorship matrix (A) as follows. A flow matrix (F) is created where $F_{(i,j)}$ represents the bibliometric flow from country i to country j. The total collaborations of a country are calculated, which can be defined as $C_{(i)}$ for country i. This can be determined by summing the values of the corresponding row in the co-authorship matrix A.

Then, the bibliometric flow from country i to country j is calculated by dividing the number of collaborations between these two countries $(A_{(i,j)})$ by the total collaborations of country i $(C_{(i)})$:

$$F_{(i.j)} = A_{(i,j)}/C_{(i)}. \quad (2)$$

The result is a flow matrix that displays the relative collaboration flow among all country pairs. This metric allows for the identification of which countries collaborate more closely in scientific production and how that collaboration is distributed globally.

Figure 10 illustrates the bibliometric flow coming from different countries. This visualization provides a graphic representation of the contribution and participation of various countries in the literature related to the researched topic.

This information is essential to understand the geographical distribution of the research and can help identify the leading regions on the subject, as well as potential international collaborations. It can also offer insight into how regional policies or trends might influence the evolution and

Fig. 6. Quartile levels by year

Table 6. Sectors where 5G technology is most applied

Sector	Reference	Qty (%)
Telecommunications	[7] [8] [14] [17] [33] [40] [48] [52]	8 (11,7)
Health	[6] [8] [9] [10] [12] [15] [17] [18] [28] [29] [30] [33] [36] [37] [47] [48] [50] [51] [52] [59] [60] [62]	22 (32,3)
Transport	[2] [3] [6] [7] [8] [13] [15] [16] [17] [22] [23] [28] [30] [35] [36] [38] [39] [40] [44] [48] [50] [51] [52] [53] [55] [58]	26 (38,2)
Logistics	[2] [7] [13] [25] [32] [46] [57]	7 (10,2)
Agriculture	[6] [13] [15] [17] [28] [30] [50]	7 (10,2)
Education	[48] [51] [59]	3 (4,4)
Physical	[1] [2] [4] [5] [6] [8] [9] [10] [12] [13] [14] [15] [17] [18] [19] [21] [22] [24] [27] [28] [29] [30] [33] [35] [37] [39] [42] [45] [48] [49] [51] [52] [53] [54] [55] [56] [59] [60] [65] [68]	40 (58,8)
Economy	[21] [59]	2 (2,9)
Programming	[4] [5] [9] [26] [35] [41] [60]	7 (10,2)

application of 5G technology and its impact on cybersecurity.

Figure 11 displays the list of countries with the highest bibliometric flows. The Figure above shows the bibliometric flow of countries, which allows us to appreciate the number of co-occurrences between them. This representation showcases international collaborations in the field of research.

Specifically, it details that the United Kingdom tops the list with six instances of international collaborations, positioning itself as the country with

the highest frequency in this respect, followed by China and France.

This analysis highlights the global interconnection in research on 5G technology and its impact on cybersecurity and may suggest a trend towards international collaboration on these critical and emerging topics. Additionally, it provides insight into how different regions are contributing and collaborating to advance knowledge in this specific field.

Raveendran and Tabet [74] place the United States as the country that contributes most to their research, followed by India. Lozano and Mateo [75] indicate that the United States is the country with the most international collaborations in their research, differing from the results of this review, but still considering the UK and France as the top contributors.

Based on the pre-existing findings and those obtained in this study, it is observed that the UK and the US are the countries that most frequently contribute to other countries, closely followed by France and India.

Therefore, it is suggested to consider these countries as primary reference points in future research in the field of 5G Technology and its impact on cybersecurity.

These nations, it seems, maintain a prominent position in the development and international collaboration on these critical topics, which might be of relevance for researchers looking to understand global trends and establish fruitful collaborations in this rapidly advancing field.

RQ5: How can papers be grouped based on the thematic similarity of their titles in the field of 5G Technology and its impact on Cybersecurity, and which themes predominate in each cluster? Similarity Metrics: Cosine similarity is examined and used to find the similarity of the papers based on term vectors.

Cosine Similarity: Cosine similarity allows determining the angle between two vectors; they will be similar if they are quite close in terms of direction and magnitude.

Cosine similarity helps measure the cosine of the angles between two vectors. The value of cosine similarity lies in the range of -1 to 1. A value of 1 indicates that the vectors are perfectly similar, and a value of -1 indicates that the vectors are exactly opposite to each other. Two papers are

Fig. 7. Heatmap of the number of papers by continent

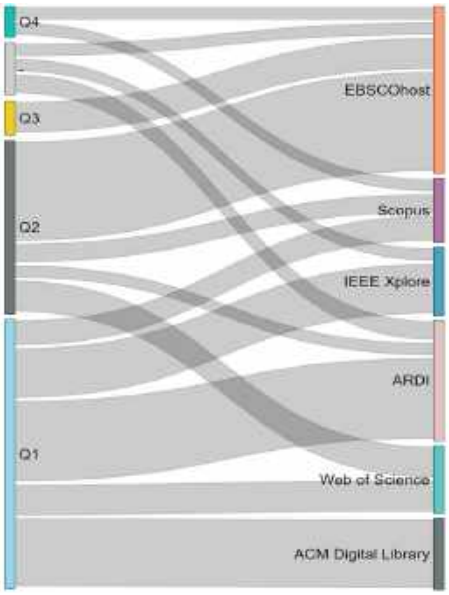


Fig. 8. Sankey diagram of quartile level by source

similar if their cosine similarity values are close to 1.

Moreover, these similarity measures are always between pairs of papers. Cosine similarity can only be calculated for vectors of similar sizes. The formula for cosine similarity for two vectors A and B is as follows:

$$\cos(\theta) = \frac{A.B}{\|A\|\|B\|}. \quad (3)$$

Here, A.B is the dot product between the two vectors, and $||A||$ and $\|B\|$ represent the magnitude of these two vectors respectively. The above formula can also be represented as follows:

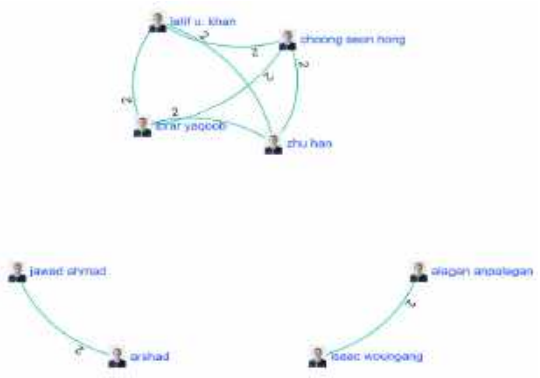


Fig. 9. Bibliometric network of co-authors

Fig. 10. Bibliometric flow of countries

$$\cos(\theta) = \frac{A.B}{\sqrt{\sum_{i=1}^{i=N} \omega_{iA}^2}\sqrt{\sum_{i=1}^{i=N} \omega_{iB}^2}}. \quad (4)$$

Here, ω_A and ω_B represent the weight or magnitude of vectors A and B along the i-th dimension, respectively, in an n-dimensional space. To find the similarity of all papers with each other, there is a shortcut model to compute all the numbers with a single command (Albrecht, Ramachandran, & Winkler, 2020, p. 127). Generalizing the formula from the previous section, we find that the similarity between paper i and paper j is as follows:

$$S_{ij} = d_i . d_j. \quad (5)$$

If one wants to use a term-document matrix, the dot product can be expressed as a summation:

$$S_{ij} = \Sigma_K D_{ik} D_{jk} = \Sigma K D_{ik} D T_{kj} = (D.D^T)_{ij}. \quad (6)$$

Therefore, this is the matrix product of the term-document matrix with its own transpose. Clustering is a machine learning task, and the techniques used in clustering can also be applied to text.

Clustering is the task of grouping data points into the same cluster, where points within the same cluster are more similar to each other than to points in different clusters. These data points can be considered as either documents or, in some cases, words.

TF-IDF is a method that assigns higher weight to rarer words, setting each element of the term-document matrix equal to the value w of multiplying the term frequency (TF) by the inverse document frequency (IDF) of each token (Kamath, Liu, & Whitaker, 2019, p. 96):

$$W = tfxidf = (1 + \log(TF_t))xlog\left(\frac{N}{n_i}\right). \quad (7)$$

Term frequency (TF) is the number of times a word appears in a document. The IDF helps to understand the importance of a word within a document.

By calculating the inverse fraction (scaled logarithmically) of the documents over the number of documents containing the term, and then taking the logarithm of that quotient, one can get a measure of how common or rare a word is across all documents. Currently, TF-IDF is the most popular weighting method, as over 80% of today's digital libraries use it.

Figure 12 illustrates four groupings of papers based on the similarity of their titles in this study. These groupings, or clusters, can help identify common themes or focuses in the existing literature on 5G Technology and its impact on cybersecurity.

By analyzing the similarity in titles, patterns or trends in the research can be uncovered, which in turn can provide valuable insight into the predominant focus areas in this field. This graphical representation aids in understanding how different works cluster together and can serve as a helpful guide to explore specific areas of interest.

The Figure groups the papers into four clusters based on the similarity of their titles. This grouping allows for the identification and categorization of works based on common themes or terminologies, thereby facilitating a clearer understanding of the main focus areas within the research on 5G Technology and its impact on cybersecurity.

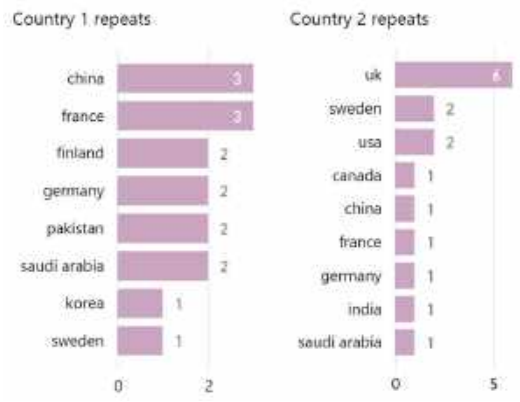


Fig. 11. Countries with the highest bibliometric flow

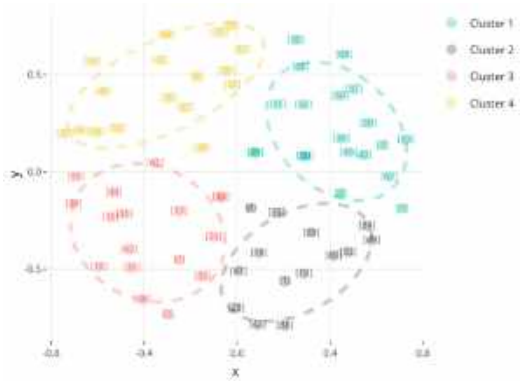


Fig. 12. Clusters of papers with title similarity

Through these clusters, an organized and structured view of the existing literature can be obtained, which could be beneficial for identifying trends, comparing approaches, and discovering potential areas that still require deeper exploration.

In their review, Raveendran and Tabet [74] opted for a more detailed classification and categorized the literature into 11 thematic clusters, focusing on aspects such as direct impacts on the environment, unrecognized rebound effects, and direct impacts on humans.

This more granular categorization allowed for a deep and specific exploration of various aspects and consequences associated with 5G technology. In contrast, the organization into 4 clusters, as shown in Figure 11, might provide a more general view but less detailed insight into the prevailing themes in the literature on 5G Technology and cybersecurity.

Both approaches have their advantages and disadvantages, and the choice between them will depend on the specific goals of the review and the level of detail researchers wish to achieve in their analysis of the existing literature.

Indeed, the organization of papers into clusters is a strategy that allows for grouping information in a way that facilitates analysis and interpretation of data.

This clustering can be done based on various criteria, such as similarity in titles, themes, methodologies, authors, among others. The way one decides to organize the clusters will depend on the focus of the research and the objectives they wish to achieve.

Therefore, in future studies, it is recommended that authors clearly define the criteria they will use to group the papers into clusters, so that this organization allows them to gain a better understanding of the state of the art and key findings in the area of study they are exploring.

It is also crucial that the selected criteria for clustering align with the research questions posed, ensuring that the organization of the data effectively contributes to answering these questions and achieving the research objectives.

5 Conclusions and Future Research

This paper presents a meticulous bibliometric and systematic review on the deployment of 5G Technology in the field of Cybersecurity, adopting the methodology proposed by Petersen. For this purpose, renowned sources such as Scopus, Web of Science, ARDI, ACM Digital Library, IEEE Xplore, and EBSCOhost were selected.

Through the application of specific equations, 13,235 studies were identified, to which meticulously designed exclusion criteria were applied. Using the Prisma diagram, the selection of 68 papers published between 2016 and 2022 was outlined. Subsequently, the selected studies were evaluated and organized using the Mendeley software.

Regarding the findings of this investigation, it stands out that the areas of greatest focus in the application of 5G technology are health, transport, and physics. The paper titled "6G Wireless Communication Systems:

Applications, Requirements, Technologies, Challenges, and Research Directions" is notably highlighted for its significant contribution to research.

Geographically, the United Kingdom emerges as the country with the highest bibliometric flow in this area of study, underscoring its prominent position in research on the intersection of 5G Technology and Cybersecurity.

This review presents two limitations that could be addressed in future research. Firstly, it was confined to six information sources, which may have narrowed the breadth of the findings.

A more expansive exploration incorporating a wider variety of databases and information sources could provide a more holistic and representative view of the current state of knowledge at the intersection of 5G Technology and Cybersecurity.

Secondly, the considered time frame, from 2016 to 2022, while pertinent to capture the most recent developments, may have excluded pioneering or foundational research conducted before this period.

Extending the temporal scope to include studies published in earlier years could unravel a deeper and contextualized understanding of the evolution of 5G Technology and its impact on Cybersecurity, thus allowing for a richer appreciation of the trends, challenges, and opportunities that characterize this dynamic and rapidly evolving field.

References

1. **Abdel-Malek, M. A., Akkaya, K., Bhuyan, A., Ibrahim, A. S. (2022).** A proxy signature-based swarm drone authentication with leader selection in 5G networks. IEEE Access, Vol. 10, pp. 57485–57498. DOI: 10.1109/ACCESS.2022.3178121.
2. **Adeel, A., Ahmad, J., Larijani, H., Hussain, A. (2020).** A novel real-time, lightweight chaotic-encryption scheme for next-generation audio-visual hearing aids. Cognitive Computation, Vol. 12, No. 3, pp. 589–601. DOI: 10.1007/s12559-019-09653-z.
3. **Afzal, R., Murugesan, R. K. (2022).** Rule-based anomaly detection model with stateful correlation enhancing mobile network security. Intelligent Automation & Soft Computing, Vol. 31, No. 3, pp. 1825–1841. DOI: 10.32604/iasc.2022.020598.
4. **Al-Heety, O. S., Zakaria, Z., Ismail, M., Shakir, M. M., Alani, S., Alsariera, H. (2020).** A comprehensive survey: Benefits, services, recent works, challenges, security, and use cases for SDN-VANET. IEEE Access, Vol. 8, pp. 91028–91047. DOI: 10.1109/ACCESS.2020.2992580.
5. **Alanazi, F., Jambi, K., Eassa, F., Khemakhem, M., Basuhail, A., Alsubhi, K. (2022).** Ensemble deep learning models for mitigating DDoS attack in software-defined network. Intelligent Automation & Soft Computing, Vol. 33, No. 2, pp. 923–938. DOI: 10.32604/iasc.2022.024668.
6. **Anagnostopoulos, N. A., Ahmad, S., Arul, T., Steinmetzer, D., Hollick, M., Katzenbeisser, S. (2020).** Low-cost security for next-generation IoT networks. ACM Transactions on Internet Technology (TOIT), Vol. 20, No. 3, pp. 1–31. DOI: 10.1145/3406280.
7. **Ansari, S., Ahmad, J., Aziz-Shah, S., Kashif-Bashir, A., Boutaleb, T., Sinanovic, S. (2020).** Chaos-based privacy preserving vehicle safety protocol for 5G connected autonomous vehicle networks. Transactions on Emerging Telecommunications Technologies, Vol. 31, No. 5, pp. e3966. DOI: 10.1002/ett.3966.
8. **Arfaoui, G., Bisson, P., Blom, R., Borgaonkar, R., Englund, H., Félix, E., Klaedtke, F., Nakarmi, P., Näslund, M., O'Hanlon, P., Papay, J., Suomalainen, J., Surridge, M., Wary, J. P., Zahariev, A. (2018).** A security architecture for 5G networks. IEEE Access, Vol. 6, pp. 22466–22479. DOI:10.1109/ACCES.218.28 27419.
9. **Basir, R., Chughtai, N. A., Ali, M., Qaisar, S., Hashmi, A. (2022).** Mode selection, caching and physical layer security for fog networks. Bulletin of the Polish Academy of Sciences Technical Sciences, Vol. 70, No. 5, pp. 1–11. DOI: 10.24425/bpast.2022.142652.
10. **Bordel, B., Alcarria, R., Robles, T., Iglesias, M. S. (2021).** Data authentication and anonymization in IoT scenarios and future 5G

networks using chaotic digital watermarking. IEEE Access, Vol. 9, pp. 22378–22398. DOI: 10.1109/ACCESS.2021.3055771.

11. **Cao, L., Lu, X., Gao, Z., Han, M., Du, X. (2020).** Multilevel security network communication model based on multidimensional control. Mathematical Problems in Engineering, Vol. 2020, pp. 1–18. DOI: 10.1155/2020/3528439.
12. **Carvalho, G. H. S., Woungang, I., Anpalagan, A., Traore, I. (2020).** When agile security meets 5G. IEEE Access, Vol. 8, pp. 166212–166225. DOI: 10.1109/ACCESS.2020.3022741.
13. **Chen, Y., Sambo, Y. A., Onireti, O., Imran, M. A. (2022).** A survey on LPWAN-5G integration: Main challenges and potential solutions. IEEE Access, Vol. 10, pp. 32132–32149. DOI: 10.1109/ACCESS.2022.3160193.
14. **Cheng, Q., Nguyen, D. N., Dutkiewicz, E., Mueck, M. (2019).** Preserving honest/dishonest users' operational privacy with blind interference calculation in spectrum sharing system. IEEE Transactions on Mobile Computing, Vol. 19, No. 12, pp. 2874–2890. DOI: 10.1109/TMC.2019.2936377.
15. **Chowdhury, M. Z., Shahjalal, M., Ahmed, S., Jang, Y. M. (2020).** 6G wireless communication systems: Applications, requirements, technologies, challenges, and research directions. IEEE Open Journal of the Communications Society, Vol. 1, pp. 957–975. DOI: 10.1109/OJCOMS.2020.3010270.
16. **Condoluci, M., Gallo, L., Mussot, L., Kousaridas, A., Spapis, P., Mahlouji, M., Mahmoodi, T. (2019).** 5G V2X system-level architecture of 5GCAR project. Future Internet, Vol. 11, No. 10, p. 217. DOI: 10.3390/fi11100217.
17. **El-Mettiti, A., Oumsis, M. (2022).** A survey on 6G networks: Vision, requirements, architecture, technologies and challenges. Ingénierie Des Systèmes D'Information, Vol. 27, No. 1. DOI: 10.18280/isi.270101.
18. **Haddad, Z., Baza, M., Mahmoud, M. M. E. A., Alasmary, W., Alsolami, F. (2021).** Secure and efficient AKA scheme and uniform handover protocol for 5G network using Blockchain. IEEE Open Journal of the Communications Society, Vol. 2, No. pp. 2616–2627. DOI: 10.1109/OJCOMS.2021.3131552.
19. **Hamamreh, J. M., Ankarali, Z. E., Arslan, H. (2018).** CP-Less OFDM with alignment signals for enhancing spectral efficiency, reducing latency, and improving PHY security of 5G services. IEEE Access, Vol. 6, pp. 63649–63663. DOI: 10.1109/ACCESS.2018.2877321.
20. **Hao, Y., Yan, X., Wu, J., Wang, H., Yuan, L. (2021).** Multimedia communication security in 5G/6G: coverless steganography based on image text semantic association. Security and Communication Networks, pp. 1–12. DOI: 10.1155/2021/6628034.
21. **Hatzivasilis, G., Soultatos, O., Ioannidis, S., Spanoudakis, G., Katos, V., Demetriou, G. (2020).** MobileTrust. ACM Transactions on Cyber-Physical Systems, Vol. 4, No. 3, pp. 1–25. DOI: 10.1145/3364181.
22. **He, Y., Huang, D., Chen, L., Ni, Y., Ma, X. (2022).** A survey on zero trust architecture: challenges and future trends. Wireless Communications and Mobile Computing, Vol. 2022, pp. 1–13. DOI: 10.1155/2022/6476274.
23. **Huang, X., Yoshizawa, T., Baskaran, S. B. M. (2021).** Authentication mechanisms in the 5g system. Journal of ICT Standarization, Vol. 9, No. 2, pp. 61–78. DOI: 10.13052/jicts2245-800X.921.
24. **Hussain, S. R., Echeverria, M., Karim, I., Chowdhury, O., Bertino, E. (2019).** 5GReasoner. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp. 669–684. DOI: 10.1145/3319535.3354263.
25. **Kamran, M. I., Khan, M. A., Alsuhibany, S. A., Ghadi, Y. Y., Arshad, A., Arif, J., Ahmad, J. (2022).** A highly secured image encryption scheme using quantum walk and chaos. Computers, Materials and Continua, Vol. 73, No. 1, pp. 657–672. DOI: 10.32604/cmc.2022.028876.
26. **Jin, X., Duan, Y., Zhang, Y., Huang, Y., Li, M., Mao, M., Li, Y. (2021).** Fast search of

lightweight block cipher primitives via swarm-like metaheuristics for cyber security. ACM Transactions on Internet Technology, Vol. 21, No. 4, pp. 1–15. DOI: 10.1145/3417296.

27. **Jinpeng, W., Teng, Z., Bo, Z., Jeremy-Gillbanks, Xin, Z. (2022).** An innovative FPGA implementations of the secure frequency hopping communication system based on the improved ZUC algorithm. IEEE Access, Vol. 10, pp. 54634–54648. DOI: 10.1109/ACCESS.2022.3176609.
28. **Kaur, M., Khan, M. Z., Gupta, S., Alsaeedi, A. (2022).** Adoption of blockchain with 5g networks for industrial IoT: recent advances, challenges, and potential solutions. IEEE Access, Vol. 10, pp. 981–997. DOI: 10.1109/ ACCESS.2021.3138754.
29. **Khan, L. U., Yaqoob, I., Imran, M., Han, Z., Hong, C. S. (2020).** 6G wireless systems: a vision, architectural elements, and future directions. IEEE Access, Vol. 8, pp. 147029–147044. DOI: 10.1109/ACCESS.2020. 3015289.
30. **Khan, L. U., Yaqoob, I., Tran, N. H., Han, Z., Hong, C. S. (2020).** Network slicing: recent advances, taxonomy, requirements, and open research challenges. IEEE Access, Vol. 8, pp. 36009–36028. DOI: 10.1109/ACCESS.2020. 2975072.
31. **Khan, M., Ginzboorg, P., Niemi, V. (2019).** Privacy preserving AKMA in 5G. Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop-SSR'19, PP. 45–56. DOI: 10.1145/3338500. 3360337.
32. **Kim, Y. E., Kim, M. G., Kim, H. (2022).** Detecting IoT botnet in 5G core network using machine learning. Computers, Materials & Continua, Vol. 72, No. 3, pp. 4467–4488. DOI: 10.32604/cmc.2022.026581.
33. **Krishnan, P., Jain, K., Jose, P. G., Achuthan, K., Buyya, R. (2021).** SDN enabled QOE and security framework for multimedia applications in 5G networks. ACM Transactions on Multimedia Computing, Communications, and Applications, Vol. 17, No. 2, pp. 1–29. DOI: 10.1145/3377390.
34. **Kwon, S., Park, S., Cho, H. J., Park, Y., Kim, D., Yim, K. (2021).** Towards 5G-based IoT security analysis against Vo5G eavesdropping. Computing, Vol. 103, No. 3, pp. 425–447. DOI: 10.1007/s00607-020-00855-0.
35. **Køien, G. M. (2021).** On threats to the 5G service based architecture. Wireless Personal Communications, Vol. 119, No. 1, pp. 97–116. DOI: 10.1007/s11277-021-08200-0.
36. **Le, T. V., Hsu, C. L. (2021).** An anonymous key distribution scheme for group healthcare services in 5G-enabled multi-server environments. IEEE Access, Vol. 9, 53408–53422. DOI: 10.1109/ACCESS.2021. 3070641.
37. **Liu, J., Ma, J., Li, J., Huang, M., Sadiq, N., Ai, Y. (2020).** Robust watermarking algorithm for medical volume data in internet of medical things. IEEE Access, Vol. 8, pp. 93939–93961. DOI: 10.1109/ ACCESS.2020. 2995015.
38. **Liu, P., Liu, B., Sun, Y., Zhao, B., You, I. (2018).** Mitigating DoS attacks against pseudonymous authentication through puzzle-based co-authentication in 5G-VANET. IEEE Access, Vol. 6, pp. 20795–20806. DOI: 10.1109/ACCESS.2018. 2826518.
39. **Liyanage, M., Ahmed, I., Okwuibe, J., Ylianttila, M., Kabir, H., Santos, J. L., Kantola, R., Perez, O. L., Itzazelaia, M. U., De-Oca, E. M. (2017).** Enhancing security of software defined mobile networks. IEEE Access, Vol. 5, pp. 9422–9438. DOI: 10.1109/ ACCESS.2017.2701416.
40. **Zaki, R. M., Wahab, H. B. A. (2021).** 4G network security algorithms: overview. International Journal of Interactive Mobile Technologies, Vol. 15, No. 16, pp.127. DOI: 10. 3991/ijim.v15i16.24175.
41. **Kim, T. W., Pan, Y., Park, J. H. (2022).** OTP-based software-defined cloud architecture for secure dynamic routing. Computers, Materials & Continua, Vol. 71, No. 1, pp. 1035–1049. DOI: 10.32604/cmc.2022. 015546.

42. **Manikandan, S., Rahaman, M., Song, Y. L. (2022).** Active authentication protocol for IoV environment with distributed servers. Computers, Materials & Continua, Vol. 73, No. 3, pp. 5789–5808. DOI: 10.32604/cmc.2022.031490.
43. **Ming, Z., Li, X., Sun, C., Fan, Q., Wang, X., Leung, V. C. M. (2022).** Sleeping cell detection for resiliency enhancements in 5G/B5G mobile edge-cloud computing networks. ACM Transactions on Sensor Networks, Vol. 18, No. 3, pp. 1–30. DOI: 10.1145/3512893.
44. **Mohammadkhan, A., Ramakrishnan, K. K., Jain, V. A. (2020).** Cleang—improving the architecture and protocols for future cellular networks with NFV. IEEE/ACM Transactions on Networking, Vol. 28, No. 6, pp. 2559–2572. DOI: 10.1109/TNET.2020.3015946.
45. **Pedone, I., Lioy, A., Valenza, F. (2019).** Towards an efficient management and orchestration framework for virtual network security functions. Security and Communication Networks, Vol. 2019, pp. 1–11. DOI: 10.1155/2019/2425983.
46. **Qayyum, A., Ahmad, J., Boulila, W., Rubaiee, S., Arshad, Masood, F., Khan, F., Buchanan, W. J. (2020).** Chaos-based confusion and diffusion of image pixels using dynamic substitution. IEEE Access, Vol. 8, pp. 140876–140895. DOI: 10.1109/ACCESS.2020.3012912.
47. **Qin, S., Tan, Z., Zhou, F., Xu, J., Zhang, Z. (2021).** A verifiable steganography-based secret image sharing scheme in 5G networks. Security and Communication Networks, Vol. 2021, pp. 1–14. DOI: 10.1155/2021/6629726.
48. **Ranaweera, P., Jurcut, A., Liyanage, M. (2022).** MEC-enabled 5G use cases: A survey on security vulnerabilities and countermeasures. ACM Computing Surveys, Vol. 54, No. 9, pp. 1–37. DOI: 10.1145/3474552.
49. **Saha, T., Aaraj, N., Jha, N. K. (2022).** Machine learning assisted security analysis of 5G-network-connected systems. IEEE Transactions on Emerging Topics in Computing, Vol. 10, No. 4, pp. 2006–2024. DOI: 10.1109/TETC.2022.3147192.
50. **Sanchez-Gomez, J., Carrillo, D. G., Sanchez-Iborra, R., Hernández-Ramos, J. L., Granjal, J., Marin-Perez, R., Zamora-Izquierdo, M. A. (2020).** Integrating LPWAN technologies in the 5G ecosystem: A survey on security challenges and solutions. IEEE Access, Vol. 8, pp. 216437–216460. DOI: 10.1109/ACCESS.2020.3041057.
51. **Sharma, S. K., Woungang, I., Anpalagan, A., Chatzinotas, S. (2020).** Toward tactile internet in beyond 5G Era: recent advances, current issues, and future directions. IEEE Access, Vol. 8, pp. 56948–56991. DOI: 10.1109/ACCESS.2020.2980369.
52. **Shokoor, F., Shafik, W., Matinkhah, S. M.** Overview of 5G & beyond security. EAI Endorsed Transactions on Internet of Things, Vol. 8, No. 30. DOI: 10.4108/eetiot.v8i30.1624.
53. **Sun, Q., Lin, K., Si, C., Xu, Y., Li, S., Gope, P. (2022).** A Secure and anonymous communicate scheme over the internet of things. ACM Transactions on Sensor Networks, Vol. 18, No. 3, pp. 1–21. DOI: 10.1145/3508392.
54. **Tang, Z., Wang, J., Li, H., Zhang, J., Wang, J. (2021).** Cognitive covert traffic synthesis method based on generative adversarial network. Wireless Communications and Mobile Computing, Vol. 2021, pp. 1–14. DOI: 10.1155/2021/9982351.
55. **Tian, F., Zhang, P., Yan, Z. (2017).** A survey on C-RAN security. IEEE Access, Vol. 5, pp. 13372–13386. DOI: 10.1109/ACCESS.2017.2717852.
56. **Wang, Y., Miao, Z., Jiao, L. (2016).** Safeguarding the ultra-dense networks with the aid of physical layer security: A review and a case study. IEEE Access, Vol. 4, pp. 9082–9092. DOI: 10.1109/ACCESS.2016.2635698.
57. **Wu, Y., Wei, D., Feng, J. (2020).** Network attacks detection methods based on deep learning techniques: A survey. Security and Communication Networks, Vol. 2020, pp. 1–17. DOI: 10.1155/2020/8872923.
58. **Xie, L., Ding, Y., Yang, H., Wang, X. (2019).** Blockchain-based secure and trustworthy internet of things in SDN-Enabled 5G-

VANETs. IEEE Access, Vol. 7, pp, 56656–56666. DOI. 10.1109/ACCESS.2019.2913682.

59. **Xu, Y., Liu, S., Chen, Y. (2022).** On the problems and countermeasures of college students' mental health and safe work under network environment. Journal of Environmental and Public Health, Vol. 2022, pp. 1–11. DOI: 10.1155/2022/2993982.
60. **Yadav, N., Pande, S., Khamparia, A., Gupta, D. (2022).** Intrusion detection system on IoT with 5G network using deep learning. Wireless Communications and Mobile Computing, Vol. 2022, pp. 1–13. DOI: 10.1155/2022/9304689.
61. **Yan, J., Du, Z., Li, J., Yang, S., Li, J., Li, J. (2022).** A threat intelligence analysis method based on feature weighting and BERT-BiGRU for industrial internet of things. Security and Communication Networks, Vol. 2022, pp. 1–11. DOI: 10.1155/2022/7729456.
62. **Yan, Z., Qian, X., Liu, S., Deng, R. (2022).** Privacy protection in 5G positioning and location-based services based on SGX. ACM Transactions on Sensor Networks, Vol. 18, No. 3, pp. 1–19. DOI: 10.1145/3512892.
63. **Yu, Z., Liu, S., Wang, W. (2022).** Dynamic threat weight of network security communication based on multisource data analysis. Wireless Communications and Mobile Computing, Vol. 2022, pp. 1–11. DOI: 10.1155/2022/6729827.
64. **Zeng, S., Chen, Y. (2018).** Concurrently deniable group key agreement and its application to privacy-preserving VANETs. Wireless Communications and Mobile Computing, Vol. 2018, pp. 1–9. DOI: 10.1155/2018/6870742.
65. **Chirieleison, C., Montrone, A., Scrucca, L. (2020).** Event sustainability and sustainable transportation: a positive reciprocal influence. Journal of Sustainable Tourism, Vol. 28, No. 2, pp. 240–262. DOI: 10.1080/09669582.2019.1607361.
66. **Zhang, X., Zhu, X., Wang, J., Bao, W., Yang, L. T. (2022).** DANCE: distributed generative adversarial networks with communication compression. ACM Transactions on Sensor Networks, Vol. 22, No. 2, pp. 1–32. DOI: 10.1145/3458929.
67. **Zhu, J., Huo, L., Ansari, M. D., Ikbal, M. A. (2021).** Research on data security detection algorithm in Iot based on K-means. Scalable Computing: Practice and Experience, Vol. 22, No. 2, pp. 149–159. DOI: 10.12694:/scpe.v22i2.1880.
68. **Zhu, Y., Du, Z. (2021).** Research on the key technologies of network security-oriented situation prediction. Scientific Programming, Vol. 2021, pp. 1–10. DOI: 10.1155/2021/5527746.
69. **Petersen, K., Vakkalanka, S., Kuzniarz, L. (2015).** Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology, Vol. 64, pp. 1–18. DOI: 10.1016/j.infsof.2015.03.007.
70. **Ramaki, A. A., Rasoolzadegan, A., Bafghi, A. G. (2018).** A systematic mapping study on intrusion alert analysis in intrusion detection systems. ACM computing surveys, Vol. 51, No. 3. DOI: 10.1145/3184898.
71. **Kitchenham, B., Brereton, P. (2013).** A systematic review of systematic review process research in software engineering. Information and software technology, Vol. 55, No. 12, pp. 2049–2075. DOI: 10.1016/j.infsof.2013.07.010.
72. **Linnenluecke, M. K., Marrone, M., Singh, A. K. (2020).** Conducting systematic literature reviews and bibliometric analyses. Australian Journal of Management, Vol. 45, No. 2, pp. 175–194. DOI: 10.1177/0312896219877678.
73. **Farooqui, M. N. I., Arshad, J., Khan, M. M. (2022).** A Layered approach to threat modeling for 5G-based systems. Electronics, Vol. 11, No. 12, pp. 1–17. DOI: 10.3390/electronics11121819.
74. **Raveendran, R., Tabet-Aoul, K. A. (2021).** A meta-integrative qualitative study on the hidden threats of smart buildings/cities and their associated impacts on humans and the environment. Buildings, Vol. 11, No. 6. DOI: 10.3390/buildings11060251.
75. **Lozano, M., Sanguino, M. (2019).** Their applications: A comprehensive analysis over

time. Sensors. https://www.mdpi.com/1424-8220/19/12/2756.

76. **Zeb, S., Mahmood, A., Hassan, S. A., Piran, M. J., Gidlund, M., Guizani, M. (2022).** Industrial digital twins at the nexus of NextG wireless networks and computational intelligence: A survey. Journal of Network and Computer Applications, Vol. 200, pp. 103309. DOI: 10.1016/j.jnca.2021.103309.
77. **Ogbodo, E. U., Abu-Mahfouz, A. M., Kurien, A. M. (2022).** A Survey on 5G and LPWAN-IoT for improved smart cities and remote area applications: From the aspect of architecture and security. Sensors, Vol. 22, No. 16. DOI: 10.3390/s22166313.
78. **Ly, A., Yao, Y. D. (2021).** A review of deep learning in 5G research: Channel coding, massive MIMO, multiple access, resource allocation, and network security. IEEE Open Journal of the Communications Society, Vol. 2, pp. 396–408. DOI: 10.1109/OJCOMS.2021.3058353.
79. **Dangi, R., Jadhav, A., Choudhary, G., Dragoni, N., Mishra, M. K., Lalwani, P. (2022).** ML-Based 5G network slicing security: A comprehensive survey. Future Internet, Vol. 14, No. 4, pp. 1–28. DOI: 10.3390/fi14040116.
80. **Ben-Henda, N. (2019).** Overview on the security in 5G phase 2. Journal of ICT, Vol. 8, No. 1, pp. 1–14. DOI: 10.13052/jicts2245-800X.811.
81. **Celik, A., Tetzner, J., Sinha, K., Matta, J. (2019).** 5G device-to-device communication security and multipath routing solutions. Applied Network Science, Vol. 4, No. 1. DOI: 10.1007/s41109-019-0220-6.
82. **Segura, D., Munilla, J., Khatib, E. J., Barco, R. (2022).** 5G early data transmission (Rel-16): Security review and open issues. IEEE Access, Vol. 10, pp. 93289–93308. DOI: 10.1109/ACCESS.2022.3203722.
83. **Olimid, R. F., Nencioni, G. (2020).** 5G network slicing: A security overview. IEEE Access, Vol. 8, pp. 99999–100009. DOI: 10.1109/ACCESS.2020.2997702.
84. **Sullivan, S., Brighente, A., Kumar, S. A. P., Conti, M. (2021).** 5G security challenges and solutions: A review by OSI layers. IEEE Access, Vol. 9, pp. 116294–116314. DOI: 10.1109/ACCESS.2021.3105396.
85. **Khan, R., Kumar, P., Jayakody, D. N. K., Liyanage, M. (2020).** A survey on security and privacy of 5G technologies: Potential solutions, recent advancements, and future directions. EEE Communications Surveys & Tutorials, Vol. 22, No. 1, pp. 196–248. DOI: 10.1109/COMST.2019.2933899.
86. **Goudarzi, A., Ghayoor, F., Waseem, M., Fahad, S., Traore, I. (2022).** A Survey on IoT-enabled smart grids: emerging, applications, challenges, and outlook. Energies, Vol. 15, No. 19. DOI: 10.3390/en15196984.
87. **Sodhro, A. H., Awad, A. I., van-de-Beek, J., Nikolakopoulos, G. (2022).** Intelligent authentication of 5G healthcare devices: A survey. Internet of Things (Netherlands), Vol. 20, pp. 100610. DOI: 10.1016/j.iot.2022.100610.
88. **Ferrag, M. A., Maglaras, L., Argyriou, A., Kosmanos, D., Janicke, H. (2018).** Security for 4G and 5G cellular networks: A survey of existing authentication and privacy-preserving schemes. Journal of Network and Computer Applications, Vol. 101, pp. 55–82. DOI: 10.1016/j.jnca.2017.10.017.
89. **Gamboa-Cruzado, J., Crisóstomo-Castro, R., Vila-Buleje, J., López-Goycochea, J. (2024).** Heart attack prediction using machine learning: a comprehensive systematic review and bibliometric analysis. Journal of Theoretical and Applied Information Technology, Vol. 102, pp. 1930–1944.

Article received on 01/11/2023; accepted on 17/03/2024.
**Corresponding author is Javier Gamboa-Cruzado.*

ISSN 2007-9737

Data Stream Classification based on an Associative Classifier

Karen Pamela López-Medina[1], Abril Valeria Uriarte-Arcia[2,*], Cornelio Yáñez-Márquez[1]

[1] Instituto Politécnico Nacional,
Centro de Investigación en Computación,
Mexico

[2] Instituto Politécnico Nacional,
Centro de Innovación y Desarrollo Tecnológico en Cómputo,
México

{auriartea, cyanezm}@ipn.mx, klopezm2021@cic.ipn.mx

Abstract. Currently, the diversity of sources generating data in a massive online manner cause data streams to become part of many real work applications. Learning from a data stream is a very challenging task due to the non-stationary nature of this type of data. Characteristics such as infinite length, concept drift, concept evolution and recurrent concepts are the most common problems that need to be addressed by data stream learning algorithms. In this work an algorithm for data stream classification based on an associative classifier is presented. This proposal combines a clustering algorithm and the Naïve Associative Classifier for Online Data (NACOD) to address this problem. A set of micro-clusters (MCs), a data structure that summarizes the information of the current data, is used instead of storing the whole data. The MCs are continually updated with the arriving data, either to create new MCs or to update existing ones. The added MCs helps to deal with concept drift. To assess the performance of the proposed model, experiments were carried out on 3 data sets commonly used to evaluate data stream classification algorithms: KDD Cup 1999, Forest Cover Type and Statlog (Shuttle). Our model achieved higher accuracies than those achieved with algorithms such as data stream version of Naïve Bayes and Hoeffding Tree, the average accuracies achieved were for KDD Cup 1999: 100%, Statlog (Shuttle): 99.01% and Forest Cover Type 70.44%.

Keywords. Data stream classification, associative classifier, concept-drift.

1 Introduction

Day by day, humans are taking advantage of knowledge that we acquire to solve our tasks more easily, one of them is known in pattern recognition as Classification [10]. With great precision and according to desired parameters we can classify experiences, people, tangible or intangible things thanks to the knowledge we have previously acquired, but what happens when suddenly the nature of the data has changed?

This phenomenon is named *concept-drift*, one of the main challenges of working with data streams [13]. The detection of categories or classes has had an impact on various sectors, both in research and in industry; as an example the companies that manage their sales on Internet.

This has been acquiring greater importance since the data that is generated every day is more and larger, so it is necessary to generate efficient tools for data analysis and decision making process over the data streams [6].

The nature of data streams is dynamic and therefore they are always evolving [2]. Given this situation it is not so easy to treat them with simple and static strategies. The main challenges we face when working with data streams are:

1. *Infinite length:* the analysed data is potentially infinite so it would be impossible to store all this information.

Table 1. Characteristics of data sets used for classification performance evaluation

Data sets			
Data set name	**Number of classes**	**Number of attributes**	**Number of instances**
KDD Cup 1999	10	41	489,795
Forest Cover Type	7	54	581,012
Statlog (Shuttle)	7	9	58,000

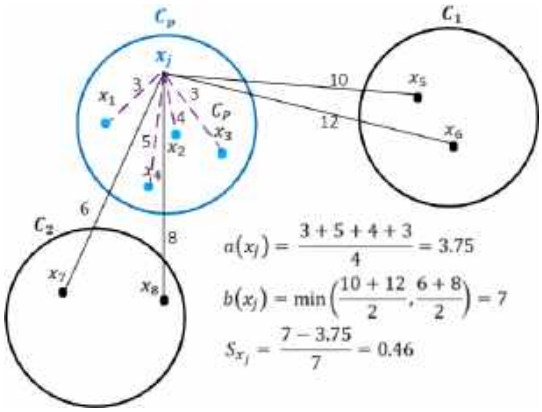


Fig. 1. Example of how the silhouette coefficient is calculated

2. *Concept-evolution:* is the phenomenon that occurs due to the presence of new unknown classes in the data.
3. *Concept-drift:* are changes in input data attribute values that indicate a change in data stream behaviour.
4. *Feature evolution:* occurs when new features appear in the data stream and initial features may disappear.

Data Mining of dynamic data stream is of great importance in the field of Machine Learning. These scenarios require adaptive algorithms that are able to process the input instances in real time, adapt to potential changes in the data, use limited computational resources, and be robust to atypical events that may occur [16]. A great deal of Machine Learning approaches have been developed to target *concept-drift detection*, such as neural networks [28, 18], active learning [12], instance based classifiers [5], Bayesian techniques [15], and ensembles classifiers [12], among others.

Data stream clustering approach is being widely applied in *data stream* scenarios. Clustering naturally adapts to speed and memory restrictions of data stream learning since it can maintain the information summarized of the cluster without the need of storing all the instances observed in a data stream.

In current literature a great deal of work addressing clustering in this type of scenarios can be found [14]. The works mentioned above provide the main ideas for the proposed solution in this document, which aims to implement a model for data stream classification that can handle concept-drift.

The rest of the paper is organized as follows: in section 2 materials and methods are described. In the following section, 3, the proposed solution and its flow chart are introduced. In section 4, the results and discussion are presented and finally in Section 5, the conclusions and proposals for future work are provided.

2 Material and Methods

This section is composed of two subsections necessary for the development of the proposed solution. In Section 2.1 the materials used in this paper are described. These materials consist of three data sets commonly used for data stream classification and the framework used to run some experiments.

In Section 2.2, the NACOD algorithm is described, which is the classifier used in this proposal. In addition, concepts such as: k-means, Silhouette coefficient and Global Hybrid Online Similarity are described.

Table 2. Example of initial training data

Number of instance	Attribute 1	Attribute 2	Attribute 3	Original Class
1	0.85	Yes	5	Class B
2	0.36	Yes	10	Class A
3	0.97	No	7	Class B
4	0.14	Yes	15	Class A
5	'?'	No	6	Class B

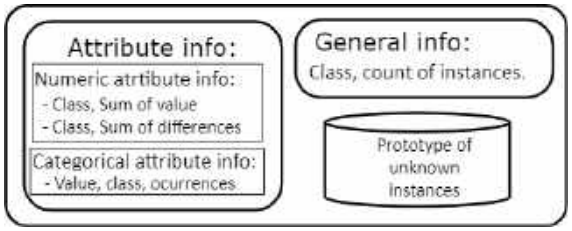


Fig. 2. Archive to store knowledge in NACOD classifier

2.1 Materials

2.1.1 Algorithms and Data Sets

This section describes the algorithms and data sets used during the experimental phase and explains the main reasons why these were selected.

MOA: to evaluate the results obtained, a comparison with other data stream classification algorithms was carried out. The MOA implementation of these algorithms was used. MOA [3] is an open source framework for massive online analysis. This framework provides several tools to deal with data stream scenarios, such as the data stream version of Naïve Bayes [3] and Hoeffding Tree [9]. The proposed methodology has been tested with three real data sets, that are often used in current works of data stream classification mainly because their size is suitable to simulate data stream scenarios. The selected data sets are:

– **KDD Cup 1999**: this is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, includes a wide variety of simulated intrusions in a military network environment. The size of the original data set has more than four million instances but commonly only 10% of them are used to save time and computational resources.

  In this work the 10% version of the data set was used, additionally only the 10 majority classes were used due to the high class imbalance produced if all the classes of this data set would be used. This decision was made since the scope of this work is not addressing the class imbalance problem.

– **Forest Cover Type**: describes seven forest cover types (classes) on a 30×30 meter cell obtained from the United States Forest Service (USFS) Region 2 Resource Information System (RIS) data. In addition, it has been used in several researches for data stream classification. The class refers to the type of forest that covers a defined area.

– **Statlog (Shuttle)**: this data set has 7 classes concerning the behavior of the pressure valve of a rocket. It has $58,000$ instances and 9 attributes, all of which are numerical. Approximately 80% of the data belongs to class 1. Therefore, the default accuracy is around 80%. Characteristics of data sets used for classification are shown in Table 1.

  The number of classes in the selected data sets varies between 7 to 10, number of attributes between 9 to 54, the minimum instance number is $58,000$ and the maximum is $581,012$. KDD Cup has 3 categorical attributes and the rest numerical. On the other hand, the Forest Cover Type and Statlog (Shuttle) attributes are integers. None of these data sets contain missing values.

2.2 Methods

The methods used in this article include clustering, specifically *k-means*, Silhouette coefficient, the associative classifier: *NACOD*, the similarity operator: *GHOS*.

2.2.1 Clustering

Clustering is an unsupervised learning task; it means that it works with data without labels. It consists of distributing a set of instances into groups according to their similarities.

General info:

(Class, count of complete instances per attribute):

Class A 2, 2, 2
Class B 2, 3, 3

Attribute info:

Categorical attribute info

(Attribute, Value, class, ocurrences):
A2, Yes, Class A, 2
A2, Yes, Class B, 1
A2, No, Class A, 0
A2, No, Class B, 2

Numeric Attribute info

(Attribute, Class, Sum of Values)
A1, Class A, 0.50
A1, Class B, 1.82
A3, Class A, 18
A3, Class B, 25

(Attribute, Class, Sum of differences $\sum_{i=1}^{n}(x_i - \tilde{x}\ )^2$)

A1, Class A, 0.0242
A1, Class B, 0.0052
A3, Class A, 13.50
A3, Class B, 2.0

Prototype

Class A: PA = (0.25, Yes, 12.5)
Class B: PB = (0.25, No, 6)

Fig. 3. Archive obtained by NACOD according to the training instances of figure 2

The major difference with respect to classification is that the number of groups is unknown before starting the learning process and the task is to create them [24]. In this work, the *k-means* algorithm will used. Next sub-section introduces theory of this algorithm.

2.2.2 K-Means

Proposed by MacQueen in 1967 [17], it consists of 2 iterative stages: the first one, an observation is assigned to the nearest cluster and in the second one cluster center is calculated according all the observations belonging to that cluster.

The algorithm does not conclude until there is no need to move any observation point to a different cluster [21]. Due to its simplicity, *k-means* [20] is one of the most used algorithms in clustering. It is known for its faster computational speed and superior performance on large data sets in comparison to other clustering techniques [29].

2.2.3 Silhouette Width Criterion

Silhouette width criterion was first defined in [25], also called Silhouette coefficient. It is based on geometrical considerations about separation and compactness of clusters and it helps to validate the micro-clusters and decide if they must be added to the model.

Consider the j^{th} element of a data set x_j which belongs to a given cluster $p \in \{1, \ldots, k\}$ and a_p be the average distance of this element to all the other elements in the cluster p. Also, let $d_{q,j}$ be the average distance from this element to all objects in other cluster, where q be each different cluster, that is $q \neq p$. Finally, let $b_{p,j}$ be the minimum $d_{q,j}$ computed over $q = 1, ..., k$, $q \neq p$, which represents the average dissimilarity of object x_j to its closest neighboring cluster. Then, the Silhouette coefficient of the individual object x_j is defined as:

$$S_{x_j} = \frac{b_{p,j} - a_{p,j}}{\max(b_{p,j}, a_{p,j})}. \quad (1)$$

Figure 1 represents an example of how Silhouette coefficient of an x_j is calculated.

2.2.4 NACOD

Recently, Villuendas-Rey *et al.* [26] introduced the Naïve Associative Classifier for Online Data (NACOD), a decision-making algorithm within the associative approach. This classifier can deal with some data complexity issues such as: missing values, hybrid and incomplete data, which significantly complicate the operation of the learning algorithms. NACOD is divided in three phases: training, classification and updating.

Training phase. It consists on storing relevant knowledge information about an initial set of labeled data.

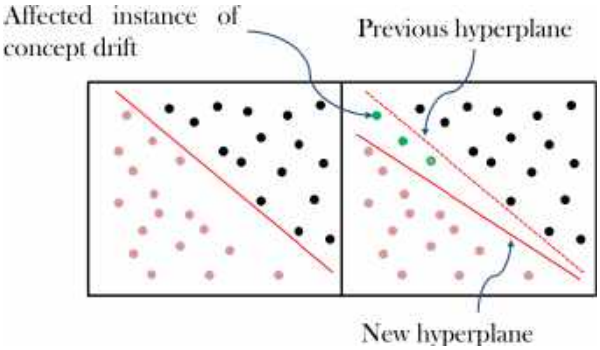


Fig. 4. Concept drift

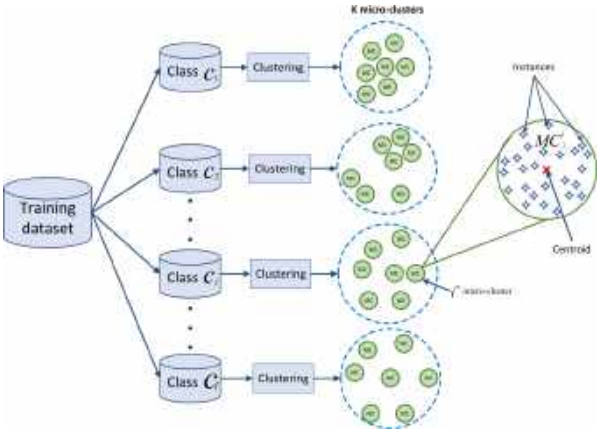


Fig. 5. Train phase

Suppose the following data sets as shown in Figure 2. The main objective in this section is to create an archive to store knowledge about known classes and attributes. Figure 2 shows how this information is handled. This archive also stores a prototype of each of the classes seen so far, like that: the class prototype is selected by using class mean for numeric attributes and class mode for categorical attributes.

For each numeric attribute, the classifier makes the sum of its complete values, and the squared sum of differences between current values and current mean. The current mean is computed by dividing the sum of complete values by the count of complete instances. The squared sum of differences allows the classifier to compute an estimation of the standard deviation of the attribute values, without the need of storing past data.

Figure 3 represents the archive corresponding to the training set described in Figure 2. NACOD uses the prototypes in the archive to compute similarities among instances. Instead of comparing the instance to classify with respect to the entire training set, it compares it with respect to the class prototypes.

Classification phase. To classify a new instance and make a decision, NACOD first computes its overall Global Hybrid Online Similarity (GHOS) with respect to the classes in the current training set (prototypes), equation 2. Then, it will assign the instance to the class with maximum similarity.

Let x be the sample to classify, let $c_j \in Y$ be a decision class, let y be the prototype of the training set of label c_j. The proposed similarity operator, named as Global Hybrid Online Similarity (GHOS) computes the overall hybrid similarity (OHS) of the instance x to the class c_j, with respect to the set of attributes $A = A_1, \ldots, A_n$:

$$\mathrm{GHOS}(x, y) = \sum_{A_i \in A} \mathrm{OHS}(x, y, A_i). \tag{2}$$

$$\mathrm{OHS}(x, y, A_i) = \begin{cases} s_c(x, y, A_i), \\ s_n(x, y, A_i), \end{cases} \tag{3}$$

where:
A_i is a categorical attribute.
A_i is a numerical attribute.

The meaning of $OHS\ (x, y, A_i)$, equation 3 had direct relationship with a relevant property of the OHS similarity immerse on the definition GHOS: OHS analyzes categorical and numeric attributes separately, without the need for codifying either of them.

For categorical attributes, s_c will return zero if the compared values of the samples are different or one of them is missing, and one if the compared values are equals, equation 5. On the other hand, for numeric attributes s_n, equation 5, uses the current standard deviation to determine if two values are close enough for considering them similar. The estimation of the standard deviation σ of numeric attribute A_i for class l_j is computed disregarding the instances with missing values for this attribute.

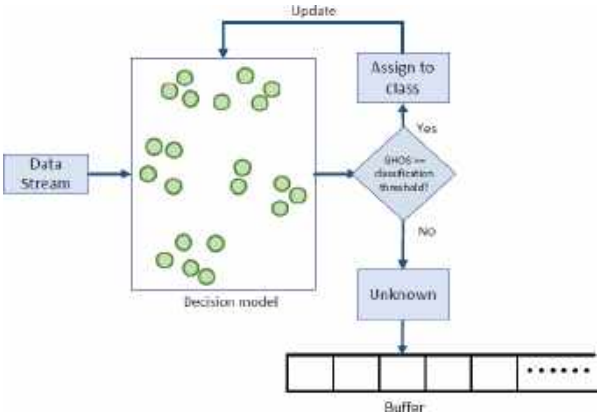


Fig. 6. Classification phase

Table 3. Initial parameters defined by the user

Initial Parameters				
A	B	C	D	E
100	20	2000	10	5

Update phase. NACOD will be classifying as new instances arrive, and then the classified instances will be used for updating the model.

Two possibilities exists: the first is that the instance belongs to a known class, and the other is that the instance does not belong to any of the existing classes. In the first situation, NACOD increases the number of instances seen so far of the corresponding class, as well as updates the maximum, minimum, sum of values and sum of squared differences for each numeric attribute, and the current values and occurrences for categorical attributes in the prototype of the corresponding class.

And in the second case, the instance will be stored by NACOD as a new class prototype; the sum of values and sum of squared differences for the new class will be initialized, as well as the values and appearances of the categorical values. NACOD pseudocode is presented in Algoritmic 1.

2.3 *Concept-Drift*

Given the non-stationary nature of data streams, the concept represented by the data that is being observed can change over time.

When this changes affect the relation between the input variables and the target variable, we are in the presence of *concept-drift* [11, 19, 12]. Formally, *concept-drift* between time point t_0 and time point t_1 can be defined as:

$$\exists X : p_{t_0}(X, y) \neq p_{t_1}(X, y), \tag{4}$$

where p_{t_0} and p_{t_1} denote the joint distributions at time t_0 and time t_1, respectively, between the set of input variables X and the target variable y. *Concept-drift* affects the decision boundaries and therefore it can reduce the performance of the learning algorithms if it is not correctly addressed. Figure 4 depicts *concept-drift*.

3 Proposed Solution

The proposal is divided into an offline phase and an online phase. During the offline phase (training) a clustering process is performed, therefore, Subsection 2.2.1 introduces the clustering algorithm used. In Subsection 2.2.3, the Silhouette coefficient was presented, which is used to determine the cohesiveness of a cluster.

The online phase uses the NACOD associative classifier to classify the arriving instances; the theory behind this classifier was presented in Section 2.2.4. Examples not described by the current model are marked as outliers and stored in a buffer to be used later to address one of the main challenges in data stream classification: *concept-drift*, explained in section 2.3, or perhaps discard them if they are outliers.

3.1 Problem Formalization

We first formalize the problem of our data sets for data stream classification here. A data stream D can be defined as an infinite sequence of multi-dimensional instances $x^1, x^2, \ldots, x^N$ arriving at time $t_1, t_2, t_3, \ldots, t_N$, where each instance is a vector of dimension n, denote by $x^i = \{x_1^i, x_2^i, \ldots, x_n^i\}$. Each instance x^i is associated with a class label y^i. Te class label $y^i \in Y$, where Y is the set of classes in the data stream $Y = \{c_1, c_2, \ldots, c_L\}$.

Algorithm 1 NACOD Classifier Algorithm

procedure NACOD(T, A, W, L, x)
  Inputs:
  T : Training set of data points with labels.
  A : Set of attributes describing each data point. $A = \{A_1, \cdots, A_n\}$
  W : Set of weights for attributes (optional). $W = \{w_1, \cdots, w_n\}$
  L : Set of possible class labels. $L = \{l_1, \ldots, l_k\}$
  x : Instance to classify.
  Training:
  Creating the archive.
  Compute and store the prototype of each label into a prototype set $P = \{p_1, \cdots, p_k\}$.
  For each numeric attribute and for each class, store its maximum and minimum values (max_i, min_i)
  For each categorical value and for each class, store its values and number of appearances.
  Classification:
  $l \leftarrow$ FindMostSimilarLabel(x, P)
  (Find the label of the prototype most similar to x)
  Updating:
  if x does not belong to any class in L **then**
    AddClassLabel(L, new label)
    UpdatePrototypeSet(P, x, new label)
    UpdateArchiveInfo(P, A)
  else
    UpdateClassInfoInArchive(P, x, L)
    UpdatePrototypeSet(P, x, GetLabelOf(x))
  end if
  Outputs:
  Class label assigned to x.
  return x
end procedure

As each instance arrives, we can collect p sequential instances into a data chunk, denoted as $D_k = \{d^{k,1}, d^{k,2}, \ldots, d^{k,t}, \ldots, d^{k,p}\}$, where $d^{k,t}$ indicates an instance that arrives at time t in data chunk k, namely $d^{k,t} = \{x^{k,t}, y^{k,t}\}$.

The training data set $D_1 = (x^{1,1}, y^{1,1}), (x^{1,2}, y^{1,2}), ..., (x^{1,t}, y^{1,t}), ..., (x^{1,p}, y1, p)$ is a set of p tuples where each $x^{1,t}$ is an input vector arriving at time t in first data chunk and $y^{1,t}$ is its target variable (class label).

3.1.1 Offline Phase

In this first phase, performed offline, that can be considered the training phase of the algorithm, a set of labeled instances will be needed to create the initial model.

As suggested in [1] an $InitNumber = 2,000$ was defined as the number of instances for the initial training set. In this stage the training data set will be divided in subsets $D_1, D_2, \ldots, D_S$ where D_k is the set of instances belonging to class c_j. A clustering algorithm is applied to each D_k to create q micro-cluster (MC) for each class. In this work, the k-means algorithm [17] will be used, as also suggested in [1]. Figure 5 depicts the general initial training process.

During the update step, which will further explained in Subsection 3.2.4, these initial micro-clusters will be adjusted to reflect the changes in the data stream. Each micro-cluster will be denoted by MC_j^i where j shows the class of the micro-cluster and i indicates the number of the micro-cluster in the class.

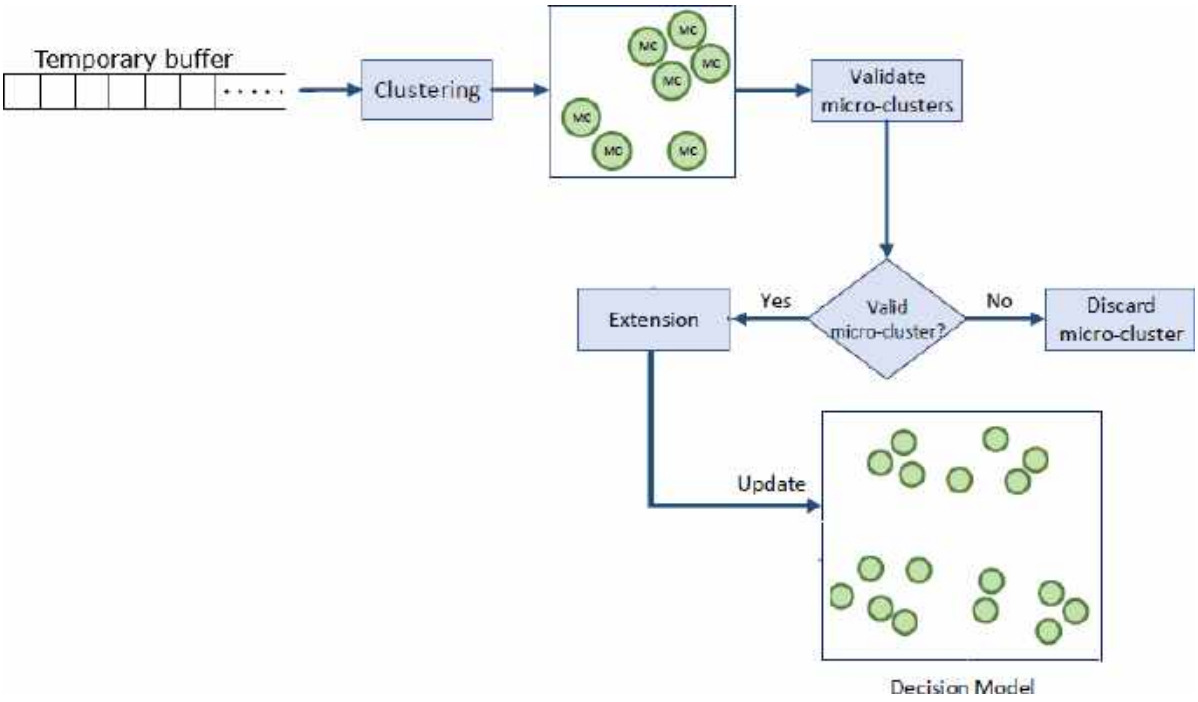


Fig. 7. *Concept-drift* detection process

Also the micro-clusters need to have an indicator of the timestamp when it was last updated. To this end, the index of the last aggregated instance will be used as the micro-cluster timestamp:

$$s_c(x, y, A_i) = \begin{cases} 0 & \text{if } \Big((x_i \neq y_i) \vee (x_i = '?') \\ & \vee (y_i = '?') \Big), \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

$$s_n(x, y, A_i) = \begin{cases} 0 & \text{if } \Big(\big(|x_i - y_i| > \sigma_j^i\big) \vee (x_i = '?') \\ & \vee (y_i = '?') \Big), \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

Each micro-cluster is described by 6 components $(N_j^i, LS, SS, SD, c_j, t_j^i)$, where N_j^i is the number of instances in the i^{th} micro-cluster of the j^{th} class; LS and SS are the linear and squared sum of the examples in the micro-cluster, respectively; SD is the sum of squared differences between the centroid of the micro-cluster and its instances; c_j is the label of the class to which the micro-cluster belongs; and t_j^i is the timestamp which value is the index of the last instance that updated the microcluster.

This representation is an extension of the one used in [8]. Using these measures it is possible to calculate other important metrics required for the implementation of this proposal, such as the standard deviation of each feature, the distance between micro-clusters, and the standard deviation of the distances between a centroid and its instances. To calculate the centroid of a micro-cluster the following equation was used, as proposed in [22]:

$$\text{Centroid}_{MC_j^i} = \frac{LS_j^i}{N_j^i}. \tag{7}$$

The equations used to calculated each of the other elements that describe a micro-cluster are the following:

$$LS_j^i = \sum_{x \in MC_j^i} x, \tag{8}$$

Fig. 8. Test-then-train or prequential with 4 blocks

$$SS_j^i = \sum_{x \in MC_j^i} (x)^2, \tag{9}$$

$$SD_j^i = \sum_{x \in MC_j^i} (x - \text{Centroid}_{MC_j^i})^2, \tag{10}$$

$$\sigma = \sqrt{\frac{SD_j^L}{N_j^L}}. \tag{11}$$

Other statistics must be calculated to use during the classification step. The NACOD classifier, uses standard deviation to calculate similarity of numeric features. Assuming the possible infinite length of a data stream, it is impossible to store all the instances that have been seen; instead summarise of the data are stored. Therefore, an estimation of the standard deviation will be calculated using the squared sum of differences. For this purpose, the squared sum of differences (SD) was calculated and stored for each micro-cluster.

3.2 Online Phase

Two possibilities exist when an instance arrives:

1. The instance is classified (correctly or incorrectly).
2. The instance is sent to a temporary buffer, because is not explained by the current model.

3.2.1 Classification

During this phase, data will be read in chunks of $2,000$ instances and the label of the new incoming instances of the data stream will be predicted. In this proposal, the NACOD algorithm [26] will be used. Unlike the original NACOD proposal [27], which only handles one standard deviation per class, our proposal uses several standard deviations per class, *i.e.* one for each micro-cluster that describes the class.

This, allows a better description of the distribution of the instances in the feature space. A classification threshold based on the GHOS operator was used to decide if a new instances should go through the classification process or should be sent to the temporary buffer. Several values were tested for this threshold, but the best results were obtained with values between $r/2$ and r, where r is the number of attributes.

The Global Hybrid Online Similarity (GHOS) is computed between the new instance and the centroids of all current micro-clusters. If the obtained value of GHOS is greater than or equal to the classification threshold proposed, the instance is classified and assigned to the class of the micro-cluster whose centroid has the maximum similarity with the instance and its information will be used to update the model. Otherwise, the instance will be store in a temporary buffer for further analysis. The general idea of the classification phase is shown in Figure 6.

3.2.2 Buffer Analysis

The instances that were not explained by the current model, *i.e.* instances stored in the temporary buffer, will be studied to determine if they are outliers or hints of the beginning of a *concept-drift*. In the second case, those instances will be used to model new micro-clusters that allow the model to incorporate this new knowledge. The process on *concept-drift* detection will be executed at the final of each data chunk of $2,000$ instances. Figure 7 shows the novel class detection process. Two possibilities exist when temporary buffer is analysed:

1. Single or multiple instances can be outliers.
2. A representative number of instances can be declared as a class extension, known as a *concept-drift*.

To determine which one of the previous cases is, is necessary to have cohesive sets of representative instances. To identify these sets, a cluster process will be applied to the temporary buffer when a considerable number of instances stored in it is reached.

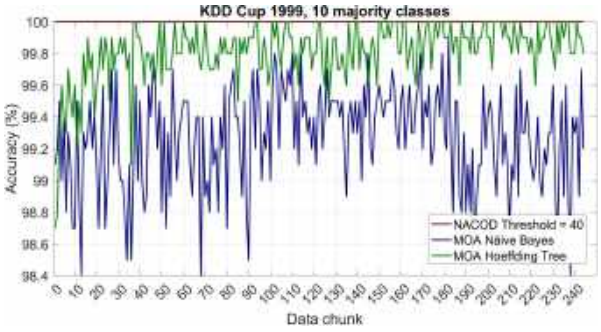


Fig. 9. KDD cup 1999 with its 10 majority classes experiment

Table 4. Accuracies averages

Using NACOD	Näive Bayes	Hoeffding Tree
100%	99.27%	99.81%
99.01%	90.02%	97.81%
70.44%	59.59%	67.78%

This number $(NumInstances)$ is a user-defined parameter of the algorithm, in the experimental section this number is called *representative number*. In this process, the *k-means* algorithm will be used again to generate these new clusters. To identify if a cluster is cohesive, a modified Silhouette coefficient will be used, as proposed in [8]:

$$\text{Silhouette} = \frac{b-a}{(a,b)}, \tag{12}$$

where b is the Euclidean distance between the new micro-cluster centroid and its closest micro-cluster centroid, and a represents the standard deviation of the distances between the instance in the new micro-cluster and its centroid. To determine if a micro-cluster is representative it has to contain a minimum number of instances.

If the new micro-cluster is valid, *i.e.* representative and cohesive, the distance between the centroid of the new micro-cluster MC_{new} and all the $MC's$ centers belonging to the decision model will be calculated. The new micro-cluster will be labeled as an extension of the same class of its closest micro-cluster, $MC_{closest}$, and its instances will be eliminated from the temporary buffer.

That means the decision boundary has changed, this explains why NACOD did not classify well. Otherwise, the temporary buffer is not modified and the instances are kept for further analysis. This may mean a *concept-drift* or *outliers*. To prevent a memory overflow, the user must define a limit number of observations to be stored in the temporary buffer.

3.2.3 Buffer Update

The temporary buffer needs to be updated: instances that have been too long in this buffer are removed. For this, the timestamp of the instances is also stored.

At the end of each data chunk, the temporary buffer is checked and the instances whose difference between their timestamp and the current timestamp is greater than the defined data chunk size are considered outdated instances and will be eliminated.

3.2.4 Model Update

When new instances $\tilde{x}$ arrive, the model needs to be updated. If an instance is explained by one of the micro-clusters, *i.e.* is correctly classified, the knowledge provided by this instance needs to be added to the corresponding micro-cluster. It means that when an instance is added to its corresponding cluster, implies that the micro-cluster statistics with greater similarity to the instance have to be updated.

Linear (LS) and squared sums (SS) are updated using equations 13 and 14, increase by one the number of instances in the micro-cluster, and the centroid needs to be recalculated using the new LS and N. The squared difference between the instance and the centroid is added to SD. The timestamp is set using the instance index. The statistics of the corresponding micro-cluster will be updated using the following equations:

$$LS_j^i = LS_j^i + \tilde{x}, \tag{13}$$

$$SS_j^i = SS_j^i + (\tilde{x})^2, \tag{14}$$

$$SD_j^i = SD_j^i + (\text{Centroid}_{MC_j^i} - \tilde{x})^2. \tag{15}$$

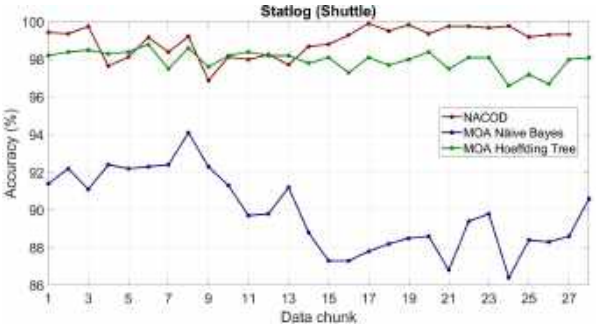


Fig. 10. Statlog (shuttle) experiment

Notice that before calculating equation 15, $\text{Centroid}_{MC_j^i}$ needs to be recalculated using the new value of LS_j^i. On the other hand, if the instance is not classified by any of the micro-clusters, it has to be added to the temporary buffer. At the end of each data chunk, the number of instances in the temporary buffer is checked to determine if it is time to analyse them and execute the *concept-drift* detection procedure.

Then a process of clustering using the *k-means* algorithm has to be executed and the Silhouette coefficient is calculated for each resulting cluster in order to assess if there are valid clusters to be added to the model. If a micro-cluster is valid, it has to be added to the model as an extension of a class. If the micro-cluster is not valid, it is discarded but the instances are kept in the temporary buffer.

4 Experimental Results

This Section explains what evaluation approach and measure metric were used. Also shows the results obtained from the experiments performed with the proposed solution presented in Section 3 and a discussion of them. Accuracy vs data chunks graphs to observe the performance of the algorithm as time goes by are provided.

Exist several approaches that are commonly used to estimate the performance of models when the data are not stationary. The objective of these approaches is to leave the last part of the data for testing and preserve the temporal order of observations although they do not make full use of all data.

In this article, we use Prequential approach, or Interleaved Test-Then-Train [7], because is one of the most common evaluation approaches in incremental data streams [11], in which first an observation (or a set of observations) is used to test and after, this observation is used to update and train the model. Figure 8 depicts Prequential approach, the data are split into 4 data splits.

As classification of data stream is not a static task, researchers evaluate over every data chuck how the classifier is working [4]. The measure implemented in this research is *accuracy*, is one of the easiest and simplest performance measure. It is determined as the ratio of correct predictions to the total number of test patterns [23].

$$\text{Accuracy} = \frac{\text{Correctly predicted values}}{\text{Total number of predictions}}. \quad (16)$$

As mentioned before, Table 3 refers to the parameters used in the experiments, those parameters must be defined by the user. The meaning of each column is explained next:

– Column A: The minimum number of instances required to start the analysis of the temporary buffer.

– Column B: The minimum number of instances to consider a MC representative.

– Column C: The Forgetting Factor, this number indicates how long an instance is kept in the temporary buffer. The forgetting process of the temporary buffer is performed by subtracting the Time Stamp of the instances in the temporary buffer to the current Time Stamp, if the result is greater that the Forgetting Factor the instance is discarded from the temporary buffer.

– Column D: The number of MC's to be generated from the temporary buffer.

– Column E: The number of MC's generated for each class in the training phase, this number refers to the *k* value for *k-means*.

The first experiment carried out was KDD Cup 1999 with its 10 majority classes. As the data set has $489,795$ instances, the data chunk length is $2,000$ and the first data chunk was used to train

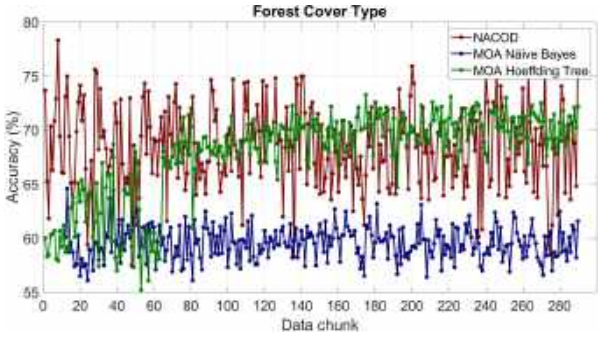


Fig. 11. Forest cover type experiment

the model, the experiment had in total 244 data chunks. Figure 9 shows 3 experimental results on this corpus. The algorithm proposed here is plotted in red, the Moa-Näive Bayes classifier is represented by the blue line and finally, the green line refers to the MOA-Hoeffding Tree.

Our proposal reaches in all the data chunks the 100% of the accuracy while Hoeffding Tree sometimes reaches the 100% accuracy and Näive Bayes never reached an 100% accuracy. It is important to mention that our proposal managed to detect *concept-drift* in 2 classes and therefore created 1 micro-cluster for the smurf class and 28 micro-clusters for the normal class.

Figure 10 corresponds to the second one experiment, which was Statlog (Shuttle), having $581,012$ observations, meaning in total 28 data chunks. The red line graphs refers to our proposal while the blue and green lines refer to the Näive Bayes and Hoeffding Tree classifiers respectively. Näive Bayes Classifier presents the lowest accuracies during the whole experiment; while out of 28 data chunks only in 6 data chunks it beats our proposal.

In Table 4 we observe the average accuracies of the whole experiments, for the second time our proposal manages to obtain the highest average accuracy compared to Näives Bayes and Hoeffding Tree. Here *concept-drift* was identified in 3 classes. 13, 6 and 4 MC's were added to the model for classes 1, 4 and 5 respectively. The main objective with this corpus was achieved with our proposal; as 80% of the data belong to class 1, the default accuracy is about 80%.

The aim here is to obtain an accuracy of 99 - 99.9%, we achieved 99.01%. Forest Cover Type data set was the last experiment carried out, its length is $581,012$ observations, if we consider each data chunk of length equal to $2,000$; it had 290 data chunks. The red line plotted corresponds to our proposal while the Näive Bayes classifier is plotted in blue and the Hoeffding Tree classifier in green.

As in the previous experiment, our proposal also identified *concept-drift* in 2 classes: class 1 and 2 ended up with 26 and 73 more MC's respectively. Näive Bayes classifier loses and it seems to be very close between Hoeffding Tree and our proposal but our proposal won again. In Table 11 the average accuracies of all the experiments are available.

5 Conclusion and Future Work

5.1 Conclusions

In this work an algorithm for classification of data streams using a classifier with an associative approach was designed, implemented and tested. This proposal combines a clustering algorithm, the Naïve Associative Classifier for Online Data (NACOD), a method to detected *concept-drift* and an updating process to incorporate the new discovered knowledge.

An extensive documentary research was carried out to study the issues related to the processing of massive data streams that change dynamically. As part of the study of current topics in data stream learning, a study of available real data sets was performed, in order to select those that were appropriate to perform the tests of the proposed algorithm. This led us to the selection of only 3 data sets for our tests.

The proposed solution was developed with the aim of classify, but also addresses *concept-drift* and noisy data. Our proposal beats the other classifiers in the evaluation measure. Because of the design of the solution, it is our belief that it is also capable of handling recurrent concepts, but given the scope, tests to verify this affirmation were not performed.

5.2 Future Work

The proposal presented in this paper fulfilled the main objective of classifying into data streams but several issues are still open to improvements to extend the scope of this proposal. Among them, we can mention the following:

– Use a different clustering algorithm to create the MC's. A clustering algorithm specially design for data stream, such as CluStream, could improve the results.
– Test the proposed solution with synthetic generated data sets.
– Design experiments to test the ability of the algorithm to handle recurring concepts and concept evolution.

Acknowledgments

The authors would like to thank the Instituto Politécnico Nacional (Secretaría de Investigación y Posgrado, Centro de Investigación en Computación and Centro de Innovación y Desarrollo Tecnológico en Cómputo), Consejo Nacional de Humanidades, Ciencia y Tecnología and Sistema Nacional de Investigadores for their support to develop this work.

References

1. **Aggarwal, C. C., Yu, P. S., Han, J., Wang, J. (2003).** A framework for clustering evolving data streams. Proceedings 2003 VLDB Conference, pp. 81–92. DOI: 10.1016/B978-012722442-8/50016-1.
2. **Bianchini, D., De-Antonellis, V., Garda, M. (2023).** A big data exploration approach to exploit in-vehicle data for smart road maintenance. Future Generation Computer Systems, Vol. 149, pp. 701–716. DOI: 10.1016/j.future.2023.08.004.
3. **Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B. (2010).** MOA: Massive online analysis. The Journal of Machine Learning Research, Vol. 11, pp. 1601–1604. DOI: 10.5555/1756006.1859903.
4. **Brzezinski, D., Stefanowski, J. (2017).** Prequential AUC: Properties of the area under the ROC curve for data streams with concept drift. Knowledge and Information Systems, Vol. 52, pp. 531–562. DOI: 10.1007/s10115-017-1022-8.
5. **Chen, D., Yang, Q., Liu, J., Zeng, Z. (2020).** Selective prototype-based learning on concept-drifting data streams. Information Sciences, Vol. 516, pp. 20–32. DOI: 10.1016/j.ins.2019.12.046.
6. **Chen, F. (2023).** Anomaly recognition method of network media large data stream based on feature learning. The International Conference on Cyber Security Intelligence and Analytics, pp. 20–28. DOI: 10.1007/978-3-031-31860-3_3.
7. **Dawid, A. P. (1984).** Present position and potential developments: Some personal views statistical theory the prequential approach. Journal of the Royal Statistical Society: Series A (General), Vol. 147, No. 2, pp. 278–290. DOI: 10.2307/2981683.
8. **de-Faria, E. R., Ponce-de-Leon-Ferreira-Carvalho, A. C., Gama, J. (2016).** MINAS: Multiclass learning algorithm for novelty detection in data streams. Data Mining and Knowledge Discovery, Vol. 30, pp. 640–680. DOI: 10.1007/s10618-015-0433-y.
9. **Domingos, P., Hulten, G. (2000).** Mining high-speed data streams. Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 71–80. DOI: 10.1145/347090.347107.
10. **Duda, R. O., Hart, P. E., Stork, D. G. (2000).** Pattern classification. Chapter Nonparametric Techniques, pp. 177–178.
11. **Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A. (2014).** A survey on concept drift adaptation. ACM computing surveys (CSUR), Vol. 46, No. 4, pp. 1–37. DOI: 10.1145/2523813.
12. **Halder, B., Hasan, K. A., Amagasa, T., Ahmed, M. M. (2023).** Autonomic active learning strategy using cluster-based ensemble classifier for concept drifts in imbalanced data stream. Expert Systems with Applications, Vol. 231, pp. 120578. DOI: 10.1016/j.eswa.2023.120578.
13. **Hewage, U. H. W. A., Sinha, R., Naeem, M. A. (2023).** Privacy-preserving data (stream) mining techniques and their impact on data mining accuracy: A systematic literature review. Artificial Intelligence Review, Vol. 56, pp. 10427–10464.

14. **Islam, M. Z., Lin, Y., Vokkarane, V. M., Yu, N. (2023).** Robust learning-based real-time load estimation using sparsely deployed smart meters with high reporting rates. Applied Energy, Vol. 352, pp. 121964. DOI: 10.1016/j.apenergy.2023.121964.
15. **Jasiński, M., Woźniak, M. (2022).** Employing convolutional neural networks for continual learning. International Conference on Artificial Intelligence and Soft Computing, pp. 288–297. DOI: 10.1007/978-3-031-23492-7_25.
16. **Korycki, Ł., Krawczyk, B. (2023).** Adversarial concept drift detection under poisoning attacks for robust data stream mining. Machine Learning, Vol. 112, No. 10, pp. 4013–4048. DOI: 10.1007/s10994-022-06177-w.
17. **MacQueen, J. (1967).** Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Vol. 1, No. 14, pp. 281–297.
18. **Mahajan, E., Mahajan, H., Kumar, S. (2024).** EnsMulHateCyb: Multilingual hate speech and cyberbully detection in online social media. Expert Systems with Applications, Vol. 236, pp. 121228. DOI: 10.1016/j.eswa.2023.121228.
19. **Mehmood, H., Khalid, A., Kostakos, P., Gilman, E., Pirttikangas, S. (2024).** A novel edge architecture and solution for detecting concept drift in smart environments. Future Generation Computer Systems, Vol. 150, pp. 127–143. DOI: 10.1016/j.future.2023.08.023.
20. **Mohanapriya, K., Sangavi, N., Kanimozhi, A., Kiruthika, V. R., Dhivya, P. (2023).** Optimized feed forward neural network for fake and clone account detection in online social networks. 2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), pp. 476–481. DOI: 10.1109/ICSCDS56580.2023.10104616.
21. **Pala, O. (2024).** Assessment of the social progress on European Union by logarithmic decomposition of criteria importance. Expert Systems with Applications, Vol. 238, pp. 121846. DOI: 10.1016/j.eswa.2023.121846.
22. **Souza, V. M., Silva, D. F., Batista, G. E., Gama, J. (2015).** Classification of evolving data streams with infinitely delayed labels. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), pp. 214–219. DOI: 10.1109/ICMLA.2015.174.
23. **Suryawanshi, S., Goswami, A., Patil, P. D., Mishra, V. (2022).** Adaptive windowing based recurrent neural network for drift adaption in non-stationary environment. Journal of Ambient Intelligence and Humanized Computing, Vol. 14, No. 8, pp. 1–15. DOI: 10.1007/s12652-022-04116-0.
24. **Tao, Z., Huang, S., Wang, G. (2023).** Prototypes sampling mechanism for class incremental learning. IEEE Access. DOI: 10.1109/ACCESS.2023.3301123.
25. **Vendramin, L., Campello, R. J., Hruschka, E. R. (2010).** Relative clustering validity criteria: A comparative overview. Statistical analysis and data mining: the ASA data science journal, Vol. 3, No. 4, pp. 209–235. DOI: 10.1002/sam.10080.
26. **Villuendas-Rey, Y., Hernandez-Castaño, J. A., Camacho-Nieto, O., Yañez-Marquez, C., Lopez-Yañez, I. (2019).** NACOD: A naïve associative classifier for online data. IEEE Access, Vol. 7, pp. 117761–117767. DOI: 10.1109/ACCESS.2019.2936366.
27. **Villuendas-Rey, Y., Rey-Benguría, C. F., Ferreira-Santiago, A., Camacho-Nieto, O., Yáñez-Márquez, C. (2017).** The naïve associative classifier (NAC): A novel, simple, transparent, and accurate classification model evaluated on financial data. Neurocomputing, Vol. 265, pp. 105–115. DOI: 10.1016/j.neucom.2017.03.085.
28. **Wu, Y. M., Chen, L. S., Li, S. B., Chen, J. D. (2021).** An adaptive algorithm for dealing with data stream evolution and singularity. Information Sciences, Vol. 545, pp. 312–330. DOI: 10.1016/j.ins.2020.07.010.
29. **Yao, B., Ling, G., Liu, F., Ge, M. F. (2023).** Multi-source variational mode transfer learning for enhanced PM2.5 concentration forecasting at data-limited monitoring stations. Expert Systems with Applications, Vol. 238, pp. 121714. DOI: 10.1016/j.eswa.2023.121714.

Article received on 07/11/2023; accepted on 14/12/2023.
**Corresponding author is Abril Valeria Uriarte-Arcia.*

ISSN 2007-9737

An IoE-SVM Based Statistical Investigation to Measure Effect of Air Pollutant Substances on Student's Attention Level

Sushil Kumar-Mahapatra[1], Binod Kumar-Pattanayak[1], Bibudhendu Pati[2,*]

[1] Siksha 'O' Anusandhan University,
Department of Computer Science and Engineering, Bhubsneswar,
India

[2] Rama Devi Women's University,
Department of Computer Science, Bhubaneswar,
India

{mohapatrasushil, patibibudhendu}@gmail.com, binodpattanayak@soa.ac.in

Abstract. The air pollutant in a specific region is mainly responsible for the respiration-related health problem. But the effect of these pollutants is also creating cognitive related issues and hence degrades the attention level of the students. As a result, their educational performance and learning ability reduce to a great extent. This study is to reveal the effect of the air pollutant on the educational outcomes of a student by measuring their attentiveness in the class using the Internet of Everything (IoE). As the effect of ambient air quality is different from region to region. This investigation is performed upon 33 schools of the Odisha region in India. An IoE-SVM Based approach is employed to evaluate the attentiveness of the student in different regions. The result shows that the PM_{10} and NO_2 are mostly affecting the student's attentiveness and hence cognitive response. The result shows that a 1% increase in PM_{10}, NO_2, and SO_2 levels would decrease the test score by 0.049%, 0.14%, and 0.18% respectively.

Keywords. Attention level, EEG signal, air pollution, academic performance, SVM, IoE, air quality index.

1 Introduction

In few decades, the air quality in different parts of the world is degrading drastically [1]. The degradation of AQI (Air Quality Index) is due to many causes, which are different from place to place depending upon the geographical position as well as through the economic and social development.

In recent years, the AQI in major cities of the world increasing rapidly due to the rapid industrialization and socio-economic developmental works [2]. This trend is not only growing in the metropolitans but also in the small cities of developing countries like India. In developing countries, the government has also taken some initiatives to urbanize small towns and villages.

As a result, industrialization comes to these rural areas of developing countries, and hence the air quality is also degrading in these areas day by day [3].

Due to the increase of air pollutants in different areas of developing countries, the people in those areas are facing severe respiratory-related problems [4].

This leads to cause of adverse health conditions due to continuous exposure to the pollutant in those areas [5]. As lower aged-group, people are more sensitive and hence more vulnerable to these pollutants that increase the cognitive defects in them [6].

Constant exposure to these air pollutants beyond their admissible level, which is prescribed by the World Health Organization (WHO) [7] and central air pollutant committee of different countries can degrade student's performance in terms of physically and academically.

In recent years, many types of research are going on to enhance educational outcomes by implementing different learning management

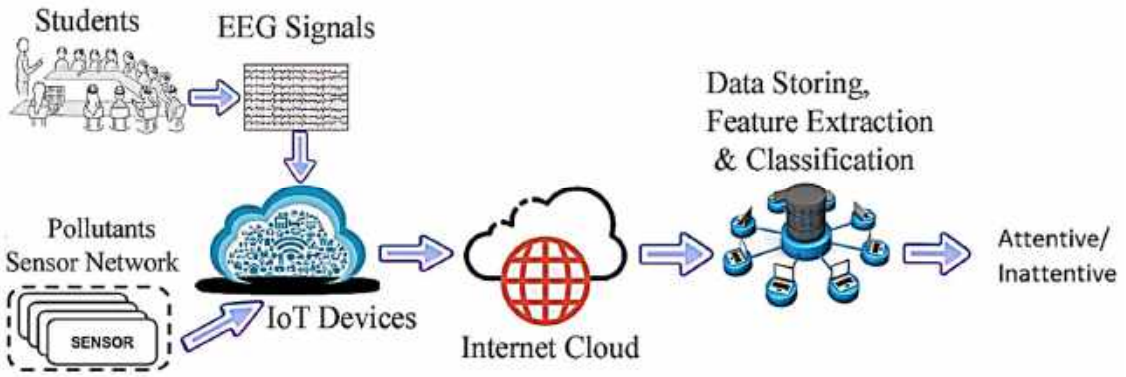


Fig. 1. Air pollutant receiving system using IoT

Table 1. Description of sensors for the air pollution detection

Name of the Sensor	Sensed Gas	O. V	Sensitivity Range
HPMA115S0 [43]	PM_{10}	5V DC	1000ug/m3
Dust sensor Module GP2Y1010AUOF [44]	$PM_{2.5}$	2.5V to 5.5V	0.5V/(100ug/m3)
MQ-7 [45]	CO	5V DC	(10-1000) ppm
GS+4NO2 [46]	NO_2	5V DC	(0-30) ppm
MQ-136 [47]	SO_2	5V DC	(1-200) ppm
MQ-131[48]	Ozone	5V DC	(10-1000) ppm

*O. V- Operating Voltage

systems using innovative learning methodology [8].

However, the impact of air pollutants on the educational outcome hasn't yet been revealed. The attendance in school and colleges, attention level, activeness, a fast response are some parameters to measure the student's performance academically. These parameters are greatly affected by air pollutants in different regions.

The air pollutants above the prescribed level can lead to an increase in absenteeism in school and an increase in mortality rate [9]. On the other hand, the education system is facing new challenges due to the intervention of Information and Communication Technology (ICT) in education [10].

The ICT has not only affected higher education but also lower grade students. Some researchers have investigated the factors that influence the transformation of a physical laboratory to a virtual laboratory using ICT in Amman [11].

In the last decade, the Internet of Things (IoT) gains its popularity due to its low-cost and easy implementation architecture. Some of the fields such as traffic control systems, health care systems, home automation, etc. have been exploited by using IoT [12-13]. Recently some research has been carried out to improve the educational system using IoT [14-15]. This study is to investigate the effect of lower and higher levels of air pollutants on the student's academic performance by using the internet of things.

This study has been carried out in 33 different schools of Odisha in India at different regions. The geographical positions of these schools are spread across the state covering the coastal area, hilly area, industrial area, etc.

1.1 Motivation

Nowadays, the impact of different environmental factors on educational outcomes has gained

Table 2. Neuro-frequency bands and their associated activities

Name of F.B.	F.R. in Hz.	V.R.in µV	R. O	Activity
Alpha	(8-13) Hz.	(30-50) µV	Parietal and occipital	In a state of consciousness, quiet or rest
Beta	(14-30) Hz.	(5-20) µV	Frontal	Conscious or alert thinking or receiving stimulation
Theta	(4-7) Hz.	Less than 100µV	Parietal and temporal	Emotional pressure, distraction, deep relaxation
Delta	(0.5-3) Hz.	(100-200) µV	Central region	Deep sleep, unconscious, anesthetized
Gamma	(31-50) Hz.	(5-10) µV	Highly localized	Selective attention

F.B.- Frequency Band
F.R.- Frequency Range
V.R.-Voltage Range
R.O- Region of occurrence

interest due to pollution in different geographical areas. As pollution level is different from place to place within a certain geographical area, hence we can see a diversified impact of this pollution level on the educational outcomes. However, the economic condition, the social condition of the students is different from place to place. Hence, a deep investigation is required to see the impact of these factors on educational outcomes.

Nowadays, IoT has gain popularity due to its low-cost architecture. The sensing ability enhances the capability of IoT to reach remote areas for data collection. These data are the vital inputs for an effective data analysis system that can measure the impact of environmental factors on the education system.

1.2 Contributions

This system is to study the effect of air pollutants in the Odisha region of India on the student's academic performance or learning outcomes. This study is also about setting up a low-cost infrastructure to get the data of different air pollutants across the Odisha region of India.

As each air pollutants has some impact on learning outcomes, this paper is to investigate and find out the statistical relationship between the air pollutant and learning outcomes.

Hence, a relationship between air pollutants and learning outcomes can be established across the country. This paper is presented as follows.

In Section 1, the introductory description of the present scenario of the air pollutant in a different geographical area with the attention level is discussed. In Section 2, past researches are presented. In Section 3, the geographical scenario of the research area is discussed.

The air pollutant and their effects on learning outcomes are presented in Section 4. In Section 5, the materials and methods are discussed which includes air pollutant receiving system using IoT, measurement, and analysis of attentiveness, etc.

In Section 6, an experimental setup is presented. The result is discussed in Section 7. Section 8, describes the concluding notes and the scope of future work.

2 Literature Review

In recent years, many researchers have identified the impact of air pollutants on hospital admission, mortality rate, cognitive response, etc. [16]. The most important air pollutants are PM_{10}, $PM_{2.5,}$ i.e. the particulate matter whose diameter is less than 10 microns and 2.5 microns respectively.

Along with these pollutants, Ozone (O_3), NOx, Sulphur Dioxide (SO_2) are also the major

Fig. 2. Neurosky brainwave sensor

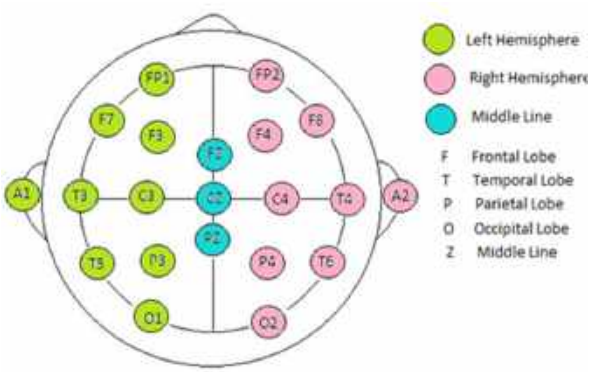


Fig. 3. Focal position of placing sensor electrodes

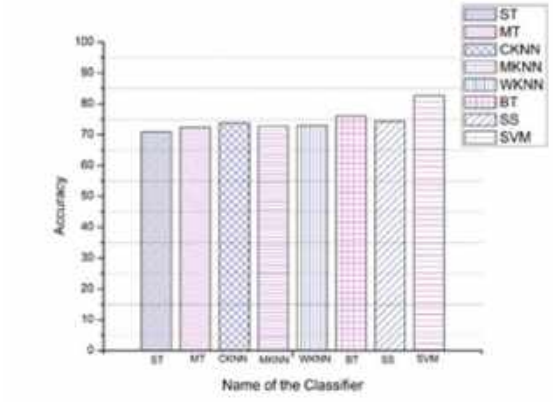


Fig. 4. Comparisons of Different Classifier for Accuracy in Predicting Attention Level

hazardous pollutants that have to be taken into consideration. The concentration of these air pollutants is also differing from season to season [17].

According to Gramsch et al., the concentration of particulate matter is greater during the autumn and winter season and the concentration of ozone is higher during the spring and summer seasons [18].

From many types of researches, it is also revealed that the effect of air pollutants is different from developing countries to developed countries [19]. As the resources like health care access are quite different from developing country to developed country, hence the research done by Arceo et al. reveals that the estimated facts for the developing countries can't be valid for the developed countries when establishing a relationship between pollution with health [20].

Therefore, the study is carried out to find out the correlation between air pollutants with the attentiveness of the students which is greatly influenced the learning outcomes. Carroll shows in their research that continuous exposure to these harmful pollutants can lead to continuous respiratory diseases and hence increases the absence rate in the school or college [21].

This also reduces their learning capability. Many researchers also trying to establish the relationship between a student's health condition to their cognitive response. To record student's cognitive responses, they have used BCI (Brain-Computer Interface) system to transfer the brain signals to a system for further analysis and actionable measures [22].

Some researchers have used EEG (electroencephalogram) signals to analyze the attentiveness of the human being [23]. But the location of the scalp from where the EEG signal has to be taken is a very critical issue that has been identified by Yaomanee et al. [24].

Some researchers also compared the utilization of EEG signal and ECG electrocardiogram) signal analyzer for the classification accuracy for attentiveness [24]. In an experiment conducted by Li et al., the level of attentiveness is measured when the subjects are given brain power-related work to be done [25].

To identify the level of attentiveness, many researchers have used different classifiers, i.e. SUM, K-Nearest Neighbors (KNN), and Naive Bayes [26]. According to Calderon et al. 2008 and Wang et al., 2009, the development of the brain

Table 3. Comparison of machine learning methods for attention prediction

Classification Method	Accuracy	Precise (%)	Recall (%)	AUC	F-Measure	G-Mean
Simple tree (ST) [57]	70.9	64.34	63.94	0.784	0.767	0.708
Medium Tree (MT) [57]	72.3	69.75	67.89	0.779	0.738	0.717
Coarse KNN (CKNN) [57]	73.8	71.69	70.26	0.825	0.816	0.734
Medium KNN (MKNN) [57]	72.7	78.91	77.36	0.893	0.838	0.723
Weight KNN (WKNN) [57]	72.9	73.67	76.14	0.836	0.819	0.725
Bagged Trees (BT) [57]	76.1	81.22	79.93	0.899	0.874	0.753
Subspace (SS) KNN [57]	74.3	78.49	76.89	0.816	0.798	0.738
SVM	**82.6**	**80.32**	**86.63**	**0.845**	**0.805**	**0.845**

and cognitive ability is greatly affected by the constant exposure to air pollutants [27-28].

Suglia et al., 2008 discussed the relation of air pollutant with the absenteeism, fatigue and neurological problem which leads to poor academic performance [29].

Some researcher (Pope, Dockery, 2006 and Russel et al., 2009) suggests that the excessive presence of PM_{10} and $PM_{2.5}$ have a negative cognitive response [30]. Graft et al., 2012 investigated the effect of O_3 on brain performance and found that if O_3 increased by 10ppb, then it leads to a decrease in productivity by 5.5% [31].

Some researchers also pointed out that the children's performance in the area of high concentration of air pollutants such as industrial areas and mining areas have poor academic performance than that of the children's in the less air pollutant area such as hilly areas (Wang et al., 2009) [32]. Zweig et al. conducted a test in California to test the effect of PM_{10}, $PM_{2.5}$, and NO_2 on academic performance [33].

The evidence collected from their research indicates that a 10% decrease in PM level would decrease the test score by 0.15% for PM_{10}, 0.22% for $PM_{2.5}$, and 0.34% for NO_2 respectively. Zaletelj and Kosir use a machine-learning algorithm to estimate the attentiveness of the student using 2D and 3D data that is collected by Kinect One sensor [34]. They have used the facial and gestures of the body to classify the attentiveness which greatly impacts the academic performance.

3 Environmental Condition of Odishga

Odisha is a state which is situated on the east coast of India [35]. As of India, it is also a densely populated area. This state is also very rich in minerals which enhances the large mining and industrialization in the state.

The coastal belt of the state is densely populated in comparison to other regions of the state as a result the vehicular emission in these areas is quite high. In recent years, the major cities have faced the worst fog situation that the state has ever experienced.

The cities like Bhubaneswar, Cuttack, Balasore, Paradeep, and Berhampur are worst affected by vehicular emission, demolition work, and constructional work.

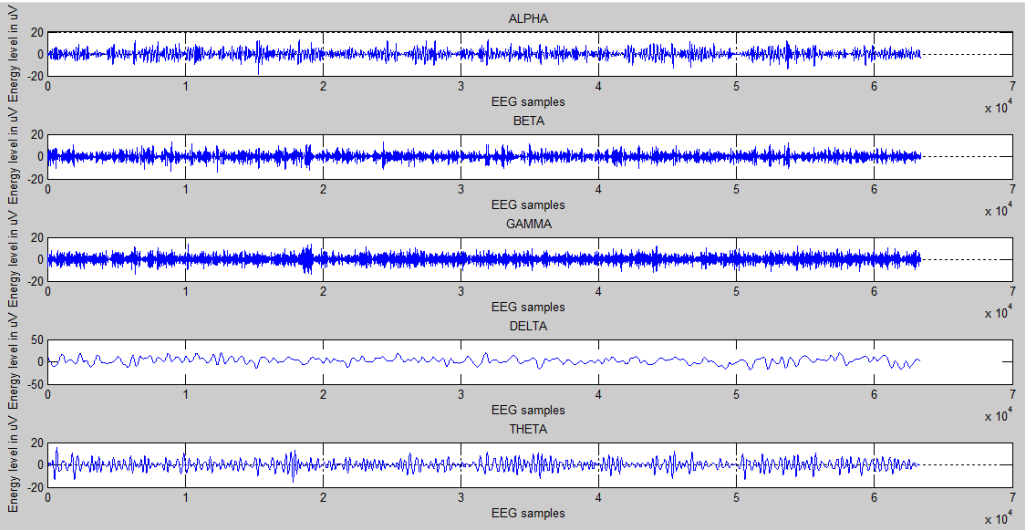


Fig. 5. Energy Level of five sub-band frequencies of EEG signal, i.e. alpha, beta, theta, delta, and gamma

The cities like Angul, Talcher, Raygada, Rourkela, and Jharsuguda which are situated in the central and northwest part of Odisha are affected by industrial pollution along with vehicular pollution.

In these cities, the temperature goes beyond 450 C in the summer season. The annual average PM_{10} level in these cities is around 300 ppm [36]. The average SO_2, NOx in these cities is around 9.2 and 25.6 respectively.

The Keonjhar and Sundargarh districts are highly affected by the PM levels as these areas are having major mines in India. So the average PM level in these areas goes beyond 400 to 500 ppm.

In a report published by the Central Pollution Control Board of India that six cities of Odisha among 200 cities of India are marked as the worst polluted [37]. Due to this reason, the number of patients suffering from cardiovascular diseases, bronchitis, and tuberculosis is increasing in these cities of Odisha.

4 Air Pollution and Its Effect on Learning Outcomes

Many researchers have identified the link between different air pollutant concentration levels, mortality and morbidity events.

According to Ponce (2012), around 4000 people die prematurely due to high concentrations of air pollutants which leads to cardiopulmonary diseases every year [38].

Ostro et al. (1999) have done a study on children under 3 to 15 years to analyze the effect of PM_{10} and Ozone [39]. They found that the children within the lower age group have lower respiratory symbols and the cognitive response is also low as compared to other age groups.

In the same manner, the higher concentration of ozone also increases hospital admission as investigated by Brunell et al. [40]. These air pollutants beyond at a certain level effects the growth of the brain, decrease the cognitive response, increases absenteeism, fatigue, etc. Due to these effects, the attention level also decreases which leads to poor academic performance.

5 Materials and Methods

In this section, a brief note about the materials used for the design of an air pollutant receiving system along with the methods used for measuring the attentiveness of the subject is discussed.

This section also reveals a detailed analysis of the attentiveness and inattentiveness of the test

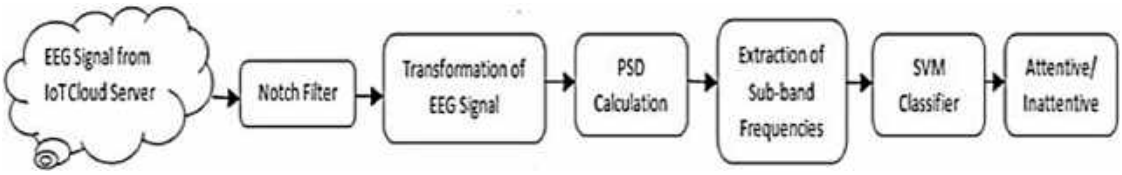


Fig. 6. Measurement of Attentiveness using EEG Signal

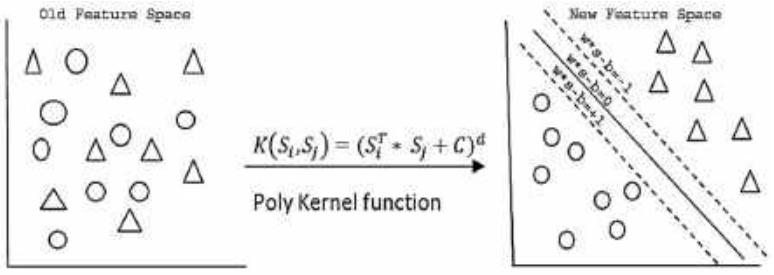


Fig. 7. SVM high dimensional feature space

subject (students) using Support Vector Machine (SVM).

5.1 Air Pollutant Receiving Station using IoT

To receive the data from the different locations, the proposed system is designed with a low cost and portable system using IoT. As recently IoT has gained popularity due to its diversity in its applications, it can be utilized to get information whenever we want it at a very low cost even from a remote location (Marques et al., 2019) [41]. In proposed system consist of a Raspberry Pi 3B+ board (IoT prototype Board) to collect the data from the different locations across the Odisha region.

To sense the different air pollutants, gas sensors are used. A detailed system for air pollutant receiving stations using IoT is shown in figure 1. The sensors are selected based on the sensitivity, power requirement and cost. To keep the initial installation cost low, some MQ series gas sensors are used (Sabuag et al., 2019) [42].

Before utilization of these sensors, all the sensors are preheated for about 48 hours and the sensors are properly calibrated in an isolation chamber as per the datasheet of the sensors. The details of the sensors are given in table 1.

The brainwave sensors are used to generate the EEG signals from the subjects and fed to the Raspberry Pi board to transmit the data to the cloud server. All these sensors are connected to the Raspberry Pi IoT prototype board (Pi, 2015) which is a SOC (System on Chip) [49].

A python program is continuously running inside the Raspberry Pi board whenever the board is connected to the power supply.

These boards and sensors require very little power usually +5VDC power and light in weight. The total system can be installed very easily at any remote location. The data collected from these sensors will be transmitted to the server by using the internet as the Raspberry Pi board has the inbuilt Wi-Fi chip embedded in it. The data is stored in a database in the server for further analysis.

5.2 Measurement of Activeness Using EEG Sensor

The attentiveness of a human being is the neurological activity of the brain [50]. Therefore, EEG sensor is a perfect candidate to measure any fluctuations that arise due to any activity of the body and brain according to Belle et al.

These sensors are capturing the electroencephalography signals called EEG signals from the brain scalps when the classes are going on to measure the attentiveness of the subject [51].

Traditionally the teacher has to identify which student is paying attention and who is not but this

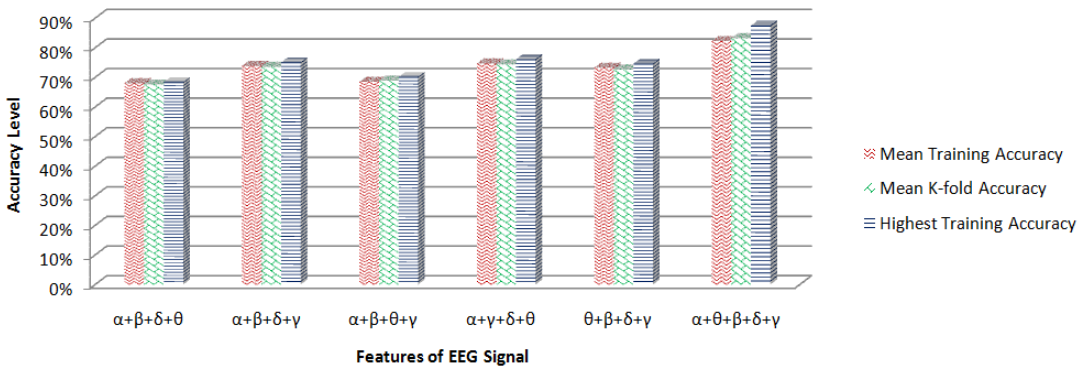


Fig. 8. Classification accuracy using different features, i.e. alpha, beta, theta, delta, and gamma

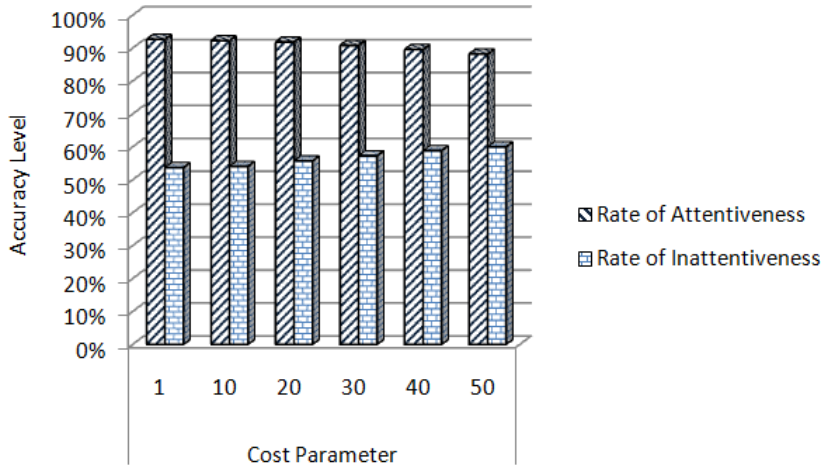


Fig. 9. shows the rate of attentiveness and rate of In-attentiveness concerning the cost function.

task is very tedious and energy-consuming. In online classes, the teacher is also unable to judge a student's attentiveness by looking at them in the video. So, whether the session is interactive or non-interactive can't be effectively determined by the teacher.

But whether the class is online or offline means face to face in a classroom, during a lecture or practical session the brain of each student generates different EEG signals in a certain band of frequency which is the base to detect the attentiveness of the student [52].

These frequency bands are associated with a certain activity of the brain and these activities can't be controlled by a normal person without training and practice. The activity associated with these frequency bands is given in Table 2 [53-54].

The proposed system consists of a brain sense device, i.e. Neurosky brainwave sensor [55], Raspberry Pi IoT prototype board, and a python program to differentiate the EEG signals. This brainwave sensor is fitted on the head of a student as shown in figure 2. A 10-20 EEG system is presented in figure 3 which is universally accepted for monitoring the EEG signals from the human brain [56].

In this system, different focal points are pointed from where the EEG signals can be monitored. Among all these focal points, two focal points in the head, i.e. FP1, FP2 result in the signal related to the attention of a person which is presented in

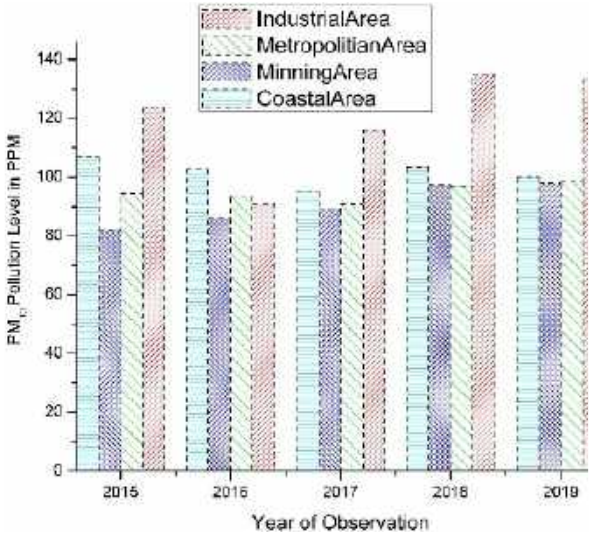


Fig. 10. Year-wise regional average of concentration of PM_{10}level in Odisha

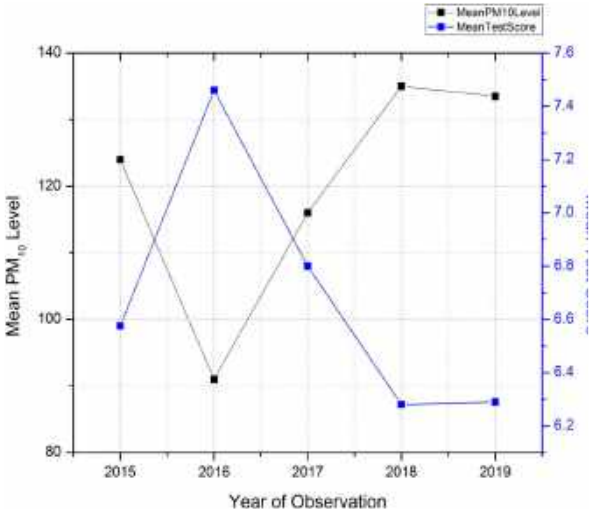


Fig.11. Comparison of test score with concentration of PM_{10} in Industrial Area of Odisha.

figure 3. The EEG signals are collected from these focal points by Neurosky Brainwave sensors and transmitted to the Raspberry Pi module through Bluetooth technology as shown in figure 1.

After collecting the EEG signals, the Raspberry Pi module segregates the signals and resends them to the IoT cloud server through Wi-Fi technology. The server receives the EEG signal values along with the air pollutant sensor value at the same time and stores them in a database for further analysis. In this work, the degree of attentiveness is classified with some well-known classifiers which is described in subsequent section.

5.3 Preprocessing of EEG Signal and Feature Selection

Before applying the SVM classifier, the features of EEG signals have to be extracted. Figure 5 represents a flow diagram for the measurement of attention level. The EEG data are collected and processed by a notch filter for the removal of the artifacts from the signal.

The signals are sampled at 512 Hz. The sampled signals are transformed into the frequency domain and the PSD (Power Spectral Density) of the sampled signal is calculated by the following formula:

$$\text{Power}(n) = \frac{s(n)s^*(n)}{N}, \tag{1}$$

where $S(n)$ is the sampled signal at the rate 512. $S*(n)$ is the complex conjugate of the signal. N is the sampling rate. As EEG signal consists of five sub-band frequencies, i.e. alpha, beta, theta, delta, and gamma, each frequency band possess a certain energy level are shown in figure 5 which is calculated as:

$$\text{Energy}_{\text{alpha}} = \sum_{f=8}^{13} \text{Power}(n), \tag{2}$$

$$\text{Energy}_{\text{beta}} = \sum_{f=14}^{30} \text{Power}(n), \tag{3}$$

$$\text{Energy}_{\text{gamma}} = \sum_{f=4}^{7} \text{Power}(n), \tag{4}$$

$$\text{Energy}_{\text{delta}} = \sum_{f=0.5}^{3} \text{Power}(n). \tag{5}$$

The extraction of sub band frequencies is then fed to the SVM for the determination of attention level. The interdependencies of all the energy sub-bands exist in between them as investigated by Hasegawa et al., the ratio of these energy levels can be taken as a feature of the attentiveness [58]:

$$\text{Rate of attentiveness} = \frac{\text{Energy}_{\text{alpha}}}{\text{Energy}_{\text{beta}}}. \tag{6}$$

All five features are calculated and are processed for classification using SVM.

5.4 SVM Classifier

In this study, an SVM classifier is used to classify whether a student is attentive or inattentive during the test. SVM creates a model to construct a hyperplane in its high dimensional feature space.

This hyperplane segregates the attentiveness or inattentiveness using the EEG signal. A polynomial kernel function is used to identify each sample from the dataset to project into the high dimensional feature space.

A parallel line is constructed on both sides of the hyperplane that differentiates the attentiveness. The SVM tries to maximize the difference between the parallel lines as shown in figure 7. The difference of parallel lines denotes the lesser the classification error rate. The kernel function is represented in equation 7:

$$K(S_i, S_j) = \left(S_i^T * S_j + C\right)^d. \quad (7)$$

After the SVM calculation and classification, to determine the accuracy of the classifier, k-fold cross-validation is performed. All the samples are partitioned into the k subset. From this subset, one subset is taken as testing data and k-1 subsets are taken as training data. To determine the accuracy, the above process is repeated for k times.

As per our research, there is no globally accepted valid dataset of EEG signal that represents the activeness or inactiveness state of mind. So, to find the exact trace of the EEG signal sub-bands for classifying attentiveness, initially a physical method is used where the students will be asked whether he is attentive or not during the class and simultaneously his EEG signals are recorded.

Then an IoT-SVM method is used to determine the attentiveness and matched with the previously recorded EEG signal which is recorded during physical method.

Two test scenarios are created in this method; one is to respond to the English phrases and respond to some questionnaire and the other is the same scenario but in presence of a noisy environment. In these two methods, the sub-bands are traced and marked.

The test subjects are asked whether they are attentive or not. If the test subject is unsure about its state of mind, then that response is discarded.

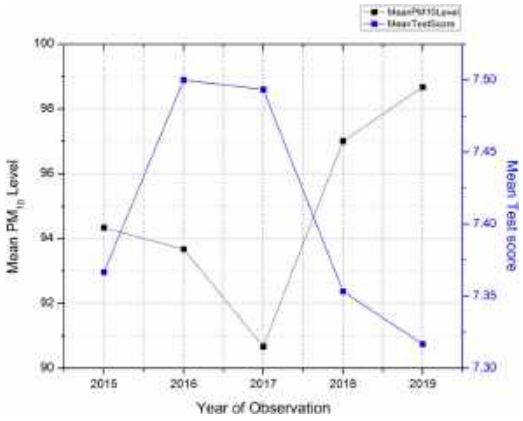


Fig.12. Comparison of test score with concentration of PM_{10} in metropolitan area of Odisha.

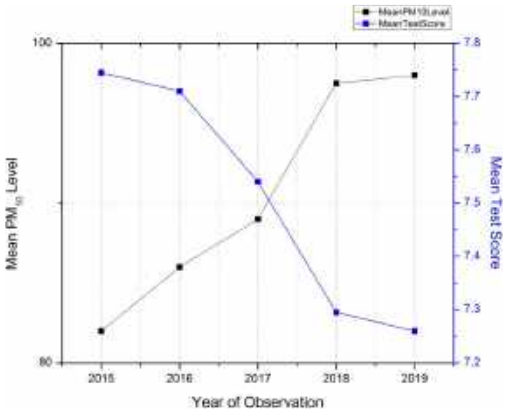


Fig.13. Comparison of test score with concentration of PM_{10} in minning area of Odisha

Likewise, a set of 3894 attentive and 2987 inattentive samples are collected. To have unbiased classification accuracy, a total of 1000 attentive and inattentive EEG signals are selected randomly for the classification.

As the attentiveness of men and women differs from each other, thus two classifications are developed for testing. Initially, a single feature is employed for the classification but the accuracy level is near about 50%. Thus, multiple features are used for classification to improve accuracy that are alpha, beta, delta, theta, gamma.

In our study, the value for k-fold is 5. Therefore, 200 samples are used for testing, and the remaining is used for training samples. The classification accuracy using different features is shown in figure 8.

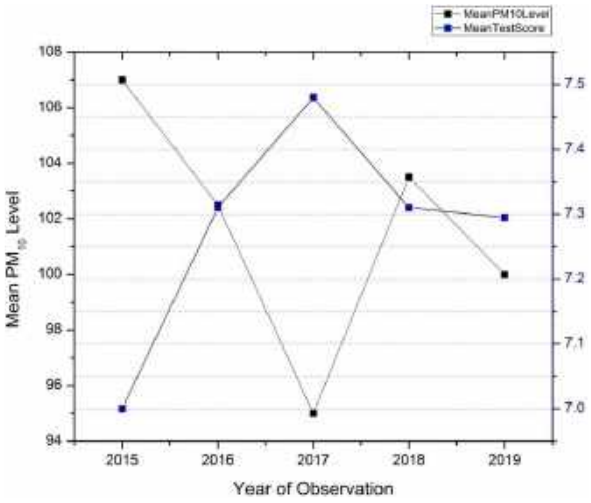


Fig. 14. Comparison of test score with concentration of PM_{10} in coastal area of Odisha

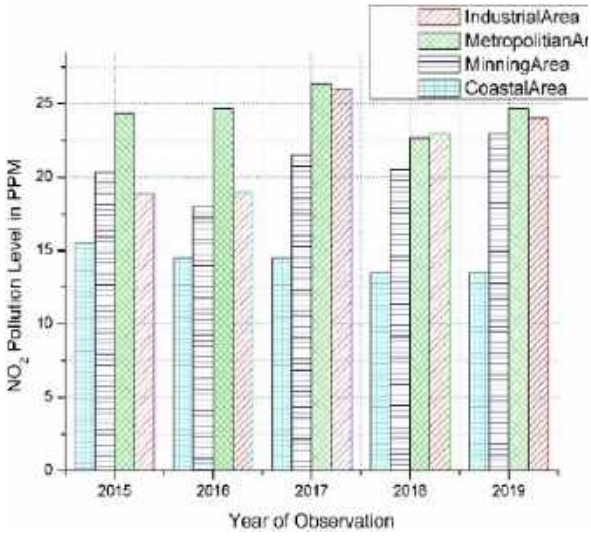


Fig.15. Year-wise regional average of concentration of NO_2 in Odisha

In the proposed study, it is also found that the rate of attentiveness and inattentiveness is also affected by the cost function of the poly kernel function denoted in equation 7.

The rate of attentiveness is decreasing as the cost function increases and the rate of inattentiveness increases as the cost function increases. Figure 9 shows the rate of attentiveness and rate of attentiveness concerning the cost function.

6 Experimental Setup

Initially, 10 cities of Odisha are selected and are categorized as Industrial area, Metropolitan area, Minning area, and Coastal area. The cities such as Angul, Rourkela are categorized as Industrial areas and Bhubaneswar, Cuttack, Sambalpur are categorized as metropolitan areas whereas Keonjhar, Sundargarh, Talcher are categorized as Mining areas and Puri, Paradeep are categorized as Coastal areas.

Around 33 schools are selected from these areas. The data is collected from these schools. While selecting the students from these schools, some points have been taken into consideration such as their family health condition, Family Income, student's medical history, the smoking habit of a parent, etc., as these factors can affect the student's performance in the test.

As there is no valid dataset of EEG signal for the evaluating attentiveness present, the EEG signals are validated manually on the test subjects preliminarily. In our test, the EEG signals and the air pollutant data are collected from 33 schools in different districts of Odisha, India. From each school, 15 male and 15 female students were selected for the test.

The selections of these students are purely random. Before experimenting, it is confirmed that any test subject has not undergone any meditation techniques to control the brain signals.

The test subjects were given the brain sensor module to wear for about 5 minutes to get familiar with the module and avoid discomfort during the test. The EEG signals were collected from each test subject during different subjects taught to them.

In between the class, the test subjects were also asked to do some tasks and read some phrases. The degree of attractiveness or inattentiveness is categorized according to the test subjects. Initially, all the test is conducted in every month. Thereafter the frequency of the test is doubled. The corresponding air pollutant levels are also measured simultaneously on such days.

The BSE (Board of Secondary Education) results from the schools are taken into consideration to validate the outcomes which is a standard examination conducted by Govt. of Odisha for the secondary school students.

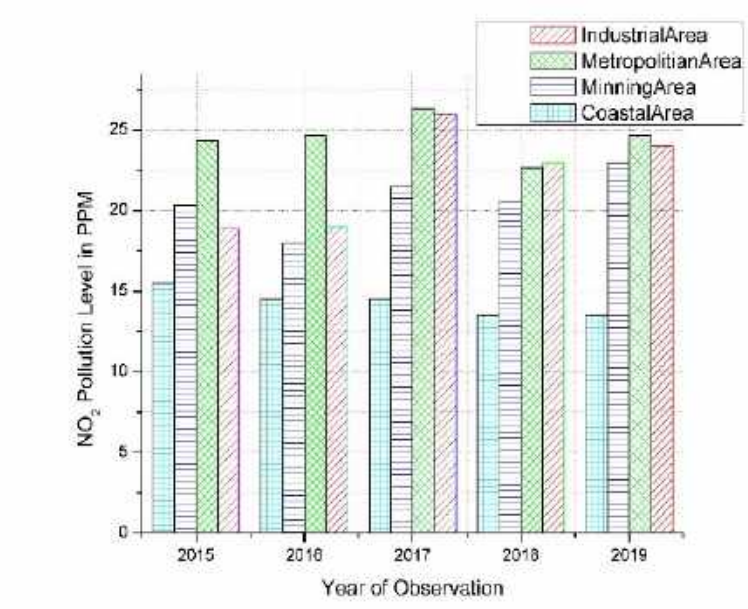


Fig.16. Year-wise regional average of concentration of SO_2 in Odisha

Table 4. Correlation between air pollutants

Pollutants	PM_{10}	$PM_{2.5}$	SO_2	NO_2	CO	O_3
PM_{10} Mean	1.00					
$PM_{2.5}$ Mean	0.568	1.00				
SO_2 Mean	0.018	0.628	1.00			
NO_2 Mean	0.728	0.868		1.00	0.876	
CO Mean	0.197	0.843		0.876	1.00	
O_3 Mean	-0.332	-0.431	-0.739	-0.573	-0.542	1.00

7 Result and Discussion

This section is divided into two subsections. One is to analyze the factors influencing attentiveness accuracy and in the second subsection, the impact of air pollutants upon the test score of the participant students is discussed. Initially, the investigation is carried upon the effect of different EEG features on attentiveness.

Different features are extracted from the EEG signal, which is collected from the IoT devices. These features are used to classify whether a student is attentive or inattentive. The effect of the performance of a student can be measured from his attentiveness.

Thus, to classify the attentiveness, the SVM classifier is used as it is found to be the best classifier among other classifiers compared in table 3. In our experimentation, we found that the accuracy level of attentiveness is optimum when all the features are taken into the consideration.

The comparison of accuracy levels is presented in figure 8. In our proposed system, it has also come into notice that as the cost parameter of the poly kernel function increases, the attentiveness rate decreases and vice versa.

After identifying the attentiveness of a student, we further investigate the effect of air pollutants on the student's attentiveness and upon their test scores.

Table 5. Concentration of PM_{10} in different cities of Odisha

Name of the City	2019		2018		2017		2016		2015	
	M. H. A	A. A	M.H. A	A. A	M. H. A	A. A	M. H. A	A. A	M. H. A	A. A
Angul	161	105	163	103	182	96	198	100	234	106
Talcher	203	94	206	96	200	91	233	94	267	93
Rourkela	274	162	265	167	339	136	217	82	278	142
Sambalpur	288	86	287	85	110	82	92	79	89	68
Sundargarh	234	102	230	99	221	87	183	78	172	71
Jharsuguda	154	103	153	106	136	91	117	87	106	82
Bhubaneswar	297	102	290	100	242	96	314	109	302	113
Cuttack	248	108	246	106	160	94	257	93	196	102
Paradeep	324	108	317	119	238	112	250	117	283	123
Puri	172	92	167	88	152	78	367	88	166	91

*M.H.A-Maximum Hourly Average
**A.A-Annual Average

Table 6. Comparison of test score for different concentration of PM_{10} in different cities of Odisha

Name of the City	Year-2019		Year-2018		Year-2017		Year-2016		Year-2015	
	PM_{10} Level	Test Score	PM_{10} Level	Test Score	PM_{10} Level	Test Score	PM_{10} Level	Test Score	PM_{10} Level	Test Score
Angul	105	7.10	103	7.12	96	7.32	100	7.14	106	7.14
Talcher	94	7.38	96	7.32	91	7.45	94	7.38	93	7.41
Rourkela	162	5.48	167	5.44	136	6.28	82	7.78	142	6.01
Sambalpur	86	7.67	85	7.72	82	7.78	79	8.01	68	8.12
Sundargarh	102	7.14	99	7.27	87	7.63	78	8.04	71	8.08
Jharsuguda	103	7.12	106	7.10	91	7.45	87	7.63	82	7.78
Bhubaneswar	102	7.14	100	7.24	96	7.32	109	7.08	113	6.84
Cuttack	108	7.14	106	7.10	94	7.38	93	7.41	102	7.14
Paradeep	108	7.14	119	6.84	112	6.94	117	6.88	123	6.55
Puri	92	7.45	88	7.78	78	8.12	88	7.78	91	7.45

*M.H.A-Maximum Hourly Average
**A.A-Annual Average

The air pollutant level of different areas across Odisha is collected from the year 2015 to 2019.

The air pollutant level data from 2015 to 2018 are collected from the State Pollution Control Board database and for 2019, the data is generated from the sensors connected with the IoT devices. The collected data is further analyzed to find out the effect of air pollutants on the student's academic performance using Wald statistical test.

In this proposed system, the impact of air pollutants concerning the test report is measured and analyzed. The test report consists of two tests. The first test is English phrase reading and responding to some queries. The second test is doing some math problems. As the first test is upon linguistic acquisition, it is greatly influenced by the feature 'delta' activity of the brain.

From the experimentation, it is found that the feature 'delta' and 'gamma' activity have more impact on attentiveness rate. In some research, it is also proposed that the feature 'theta' and 'beta' activity have minimal effect on the attentiveness

Table 7. Concentration of NO_2 in different cities of Odisha

Name of the City	2019		2018		2017		2016		2015	
	M.H.A	A.A	M.H.A	A.A	M.H.A	A.A	M.H.A	A.A	M.H.A	A.A
Angul	34	26	35	25	32	25	31	24	33	25
Talcher	33	31	34	29	37	31	29	24	36	27
Rourkela	36	22	32	21	43	27	35	14	32	13
Sambalpur	45	23	43	21	27	20	19	17	20	17
Sundargarh	26	15	23	12	20	12	21	12	24	14
Jharsuguda	34	20	31	18	24	20	24	20	23	19
Bhubaneswar	52	18	50	16	36	25	61	25	43	24
Cuttack	46	33	42	31	42	34	38	32	40	32
Paradeep	24	13	21	12	25	14	22	14	27	15
Puri	26	14	25	15	29	15	31	15	33	16

*M.H.A-Maximum Hourly Average
**A.A-Annual Average

Table 8. Concentration of SO_2 in different cities of Odisha

Name of the City	2019		2018		2017		2016		2015	
	M.H.A	A.A	M.H.A	A.A	M.H.A	A.A	M.H.A	A.A	M.H.A	A.A
Angul	21	10	20	9	18	9	14	9	14	10.8
Talcher	17	11	15	10	13	10	12	10	12	8.5
Rourkela	22	14	20	14	26	21	30	12	15	5.1
Sambalpur	41	6	39	5	8	4	5	4	5	2.5
Sundargarh	20	8	18	8	16	7	14	6.2	14	7
Jharsuguda	25	11	24	10	20	11	18	13	16	12
Bhubaneswar	16	2	15	2	21	2	25	2	24	2
Cuttack	10	5	9	4	7	5	7	4	6	2
Paradeep	36	21	35	19	52	20	43	22	40	21
Puri	7	3	5	2	5	2	16	2	7	2

*M.H.A-Maximum Hourly Average

rate. But to get better classification accuracy, all features must be taken into consideration. Figure 9 shows the effect of rate of attentiveness and inattentiveness concerning cost parameters. It shows that the rate of inattentiveness is about 50%, which means there are still more factors exist that influence the inattentiveness.

The regions of the state are categorized into four groups', i.e. Metropolitan region, the Mining region, the Industrial region, and the coastal region. Table 5 shows the annual average level of PM_{10} and the maximum hourly average level of PM_{10}. The test score for the math and text reading measures the attentiveness of the students.

The test score is calculated on that day when the concentration of PM_{10} level is around the annual average level. Table 6 shows the comparative data of PM_{10} concentration with the test scores of different cities of Odisha. The mean PM_{10} level of different geographical areas for the year 2015-2019 is presented in figure 10.

The comparative graphical representation of the mean PM_{10} level with the mean test score for different regions is presented from figure 11 to

Table 9. Statistical description of pollutant and test score

Variables	Year-2019	Year-2018	Year-2017	Year-2016	Year-2015
Mean of PM_{10}	106.20	106.90	96.30	92.70	99.10
Mean of NO_2	21.50	20	22.30	19.70	20.20
Mean of SO_2	9.1	8.3	9.1	8.42	7.29
Standard deviation of PM_{10}	19.81	23.1922	16.7003	12.8413	23.1250
Standard deviation of NO_2	6.56	6.6833	7.3643	6.4127	6.403
Standard deviation of SO_2	5.37	5.3965	6.7733	6.2136	6.1075
Mean of Test Score	7.07	7.093	7.36	7.513	7.25
Standard deviation of Test Score	0.59	0.64	0.49	0.39	0.66

Table 10. Effect of pollutant on test score

Pollutant	(1)	(2)	(3)	(4)	(5)
PM_{10}	-0.328	-0.738[a]			
NO_2	-0.846		-2.024[b]		
SO_2	-0.763			-1.998[b]	
O_3	0.372				0.284

figure 14 respectively. Likewise, the mean NO_2 level and SO_2 level for a different region of Odisha from the year 2015 to 2019 is presented in figure 15 and figure 16 respectively. The annual average level of PM_{10} in the cities is between 86 to 162 ppm (parts per million) for the year 2019 presented in Table 5.

Table 6 shows the mean test score for different areas for the year 2019. The test score for the year 2015 to 2018 is taken from the database created from the IoT devices in the year 2019 when the air pollutant level is nearer to that level.

Table 7 and Table 8 represent the concentration of NO_2 and SO_2 for a different region of Odisha. The test score is also calculated for the other air pollutants. Some of the pollutants do not have any significant impact on the test score thus excluded from the experiment like ozone. According to the study, the concentration of ozone increases in the spring and summer seasons.

In those days, the result shows a negative impact on the test score as expected but does not have a significant impact on test score [59]. During the experimentation, it is also come to notice that there is a correlation between the air pollutants. The effect of all air pollutants except ozone has a significant impact on the test score [60].

The correlation table for air pollutants is shown in table 8. In this work, the test score is measured keeping one air pollutant as a variable at a time and others as explanatory variables.

For experimentation, we have used all the major air pollutants to measure the effectiveness on the test score. A Wald test is performed for statistical analysis which reveals that all the four air pollutants, i.e. PM_{10}, NO_2, SO_2 have a significant impact on attentiveness as well as on the test score.

As we know that Wald test is used to investigate the degree of significance level for an independent variable over a model, we have used it to investigate the significance level of different air pollutant on test score individually and then collectively their influence on the test score.

The statistical analysis is presented in table 10. The significance of all the air pollutants to the test score is presented in column (1). From column (2) to column (5), the significance level of an individual air pollutant is shown. Here it is found that the Ozone has no impact on the test score.

Thus, data collected for Ozone have been discarded and is eliminated from the analysis part. However, the PM_{10} is the only air pollutant that has a statistically significant effect on the test score as

shown in table 9. Thereafter it is also found that PM_{10}, NO_2, SO_2 have a statistically significant effect on the test score but on the other hand, Ozone shows a positive sign which reflects that it does not affect the test score statistically.

The magnitude of effect on test score by the coefficients is calculated and found that an increase in the level of concentration of PM_{10} in standard deviation results in 0.028 points decreases in test score. In the same way, an increase in one standard deviation of NO_2 and SO_2 would result in a decrease in test scores by 0.08 points and 0.104 points out of 10 points respectively.

A relevant statistically elasticity measurement is also calculated here for those who do not familiar with the measurement units of air pollutants and the findings are: for 1% increase in PM_{10}, NO_2 and SO_2 level would decrease the test score by 0.049%, 0.14%, and 0.18% respectively.

This study also reveals that the test score is heavily affected by PM_{10} when the test is conducted upon below grade 5 students. The effect is quite low when considering relatively higher-grade students. In some cases, when the concentration level of PM_{10} is beyond 250 ppm, the test score for math decreases by 2.8 percent for every 10 units increase in concentration level.

Similarly, the other pollutant like NO_2 and CO is closely associated with the concentration of level of PM_{10}. All the results show that there is a negative nonlinear relationship between the test score and the concentration of air pollutants.

The significance level of different air pollutants is given on table 10 using Wald statistics. This clearly shows that the PM_{10} concentration level has the most significant effect on the test score. The joint significance value using Wald statistics is 39.87. From the statistical analysis, it has found that all the air pollutant has some effect on the test score.

8 Conclusion and Future Work

This study reveals a clear relationship between the air pollutant and the test score. From the experimentation, it is clear that the test score decreases due to a lack of attentiveness in presence of a higher concentration of air pollutants. The PM_{10} level is the most significant effect on the test score of a student while NO2 and SO2 have some effect on the test score. In this work, it is also revealed that the measurement of attentiveness depends upon some sub-bands of EEG signal but the measurement of inattentiveness depends upon some other factors that will be a future case of study.

The air pollutants are not only responsible for creating health problems but also degrading the performance of the student. The govt. along with the people of each area should take an effective measure to reduce the air pollution level, which will reduce the effect of long-term cognitive problems in the future generation of the country.

Cutting-edge technology like block chain, cloud computing, artificial intelligence can be utilized to securely investigate the real-time data in a virtual class environment for further analysis and to create a future road map to reduce the air pollutant. Furthermore, the effect of sound pollution and water pollution on the cognitive response and attentiveness of the student would be investigated in future work.

Acknowledgments

The authors would like to thank Siksha 'O' Anusandhan University for providing facilities. We also thank the participant schools for giving us opportunities to explore the cognitive response of students in different pollution levels. The author(s) also acknowledge the support of the State Pollution Control Board, Odisha, India.

References

1. **Manisalidis, I., Stavropoulou, E., Stavropoulos, A., Bezirtzoglou, E. (2020).** Environmental and health impacts of air pollution: a review. Frontiers in Public Health, Vol. 8. DOI: 10.3389/fpubh.2020.00014.
2. **Awopetu, M. S., Aribisala, J. O. (2019).** Air quality index as a tool for monitoring environmental degradation and health implications. Air Pollution XXVII, Vol. 236, pp. 9–11. DOI: 10.2495/air190021.

3. **Cohen, A., Anderson, H., Ostro, B., Krzyzanowski, M., Künzli, N., Gutschmidt, K. (2004).** Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors. pp. 1353-1434.
4. **Lelieveld, J., Klingmüller, K., Pozzer, A., Pöschl, U., Fnais, M., Daiber, A., Münzel, T. (2019).** Cardiovascular disease burden from ambient air pollution in Europe reassessed using novel hazard ratio functions. European Heart Journal, Vol. 40, No. 20, pp. 1590–1596. DOI: 10.1093/eurheartj/ehz135.
5. **Hoek, G., Krishnan, R. M., Beelen, R., Peters, A., Ostro, B., Brunekreef, B., Kaufman, J. D. (2013).** Long-term air pollution exposure and cardio- respiratory mortality: a review. Environmental Health, Vol. 12, No. 1. DOI: 10.1186/1476-069x-12-43.
6. **Buka, I., Koranteng, S., Osornio-Vargas, A. R. (2006).** The effects of air pollution on the health of children. Pediatrics and Child Health, Vol. 11, No. 8, pp. 513–516. DOI: 10.1093/pch/11.8.513.
7. **World Health Organization (2005).** Who air quality guidelines for particulate matter, ozone, nitrogen dioxide and sulfur dioxide. apps.who.int/iris/bitstream/handle/10665/69477/WHO_SDE_PHE_OEH_06.02_eng.pdf.
8. **Mahapatra, S. K., Pattanayak, B. K., Pati, B. (2020).** A survey on IoT issues, methods and its implication in higher education. TEST Engineering and Management, Vol. 83, No. May - June, pp. 5548—5568.
9. **Pope, C. A., Turner, M. C., Burnett, R. T., Jerrett, M., Gapstur, S. M., Diver, W. R., Krewski, D., Brook, R. D. (2015).** Relationships between fine particulate air pollution, cardiometabolic disorders, and cardiovascular mortality. Circulation Research, Vol. 116, No. 1, pp. 108–115. DOI: 10.1161/circresaha.116.305060.
10. **Esteban, R. F. C., Casildo-Bedén, N., Dominguez-Lara, S., Rodríguez, D. L., Caycho-Rodríguez, T. (2020).** Effectiveness of an ICT-based intervention to support the management of the teaching-learning process in peruvian universities. Psychology and Education Journal, Vol. 57, No. 7, pp. 71–84. DOI: 10.17762/pae.v57i7.71.
11. **Abukishk, A. A., Alkhasawneh, S. (2020).** An investigation of factors influencing physics teachers' intention to use virtual laboratory at the international schools in Amman. Problematica Educacional, Vol. 57, No. 5, pp. 44–58. DOI: 10.17762/pae.v57i5.44.
12. **Biswal, A. K., Singh, D., Pattanayak, B. K. (2020).** IoT-based voice-controlled energy-efficient intelligent traffic and street light monitoring system. Green Technology for Smart City and Society, Lecture Notes in Networks and Systems, Vol. 151, pp. 43–54. DOI: 10.1007/978-981-15-8218-9_4.
13. **Rath, M., Pattanayak, B. (2019).** Technological improvement in modern health care applications using internet of things (IoT) and proposal of novel health care approach. International Journal of Human Rights in Healthcare, Vol. 12, No. 2, pp. 148–162. DOI: 10.1108/ijhrh-01-2018-0007.
14. **Ramlowat, D. D., Pattanayak, B. K. (2019).** Exploring the internet of things (IoT) in education: a review. Information Systems Design and Intelligent Applications, Advances in Intelligent Systems and Computing, Vol. 863, pp. 245–255. DOI: 10.1007/978-981-13-3338-5_23.
15. **Mahapatra, S. K., Pattanayak, B. K., Pati, B. (2020).** Flip learning: a novel IoT-based learning initiative. Intelligent and Cloud Computing, peruvian universities, Vol. 153, pp. 59–67. DOI: 10.1007/978-981-15-6202-0_7.
16. **Zhang, Y., Wang, Z., Cao, Y., Zhang, L., Wang, G., Dong, F., Deng, R., Guo, B., Zeng, L., Wang, P., Dai, R., Ran, Y., Lyu, W., Miao, P., Su, S. (2021).** The effect of consecutive ambient air pollution on the hospital admission from chronic obstructive pulmonary disease in the Chengdu region, China. Air Quality, Atmosphere and Health, Vol. 14, No. 7, pp. 1049–1061. DOI: 10.1007/s11869-021-00998- 9.
17. **Cichowicz, R., Wielgosiński, G., Fetter, W. (2017).** Dispersion of atmospheric air pollution in summer and winter season. Environmental Monitoring and Assessment, Vol. 189, No. 12. DOI: 10.1007/s10661-017-6319-2.

18. **Gramsch, E., Cerecedabalic, F., Oyola, P., Vonbaer, D. (2006).** Examination of pollution trends in Santiago de Chile with cluster analysis of pm10 and ozone data. Atmospheric Environment, Vol. 40, No. 28, pp. 5464–5475. DOI: 10.1016/j.atmosenv.2006.03.062.
19. **Health Effects Institute (2019).** State of global air, special report. www.stateofglobalair.org/sites/default/files/soga_2019_report.pdf.
20. **Arceo, E., Hanna, R., Oliva, P. (2016).** Does the effect of pollution on infant mortality differ between developing and developed countries? evidence from Mexico City. The Economic Journal, Vol. 126, No. 591, pp. 257–280. DOI: 10.1111/ecoj.12273.
21. **Carroll, H. C. M. (2010).** The effect of pupil absenteeism on literacy and numeracy in the primary school. School Psychology International, Vol. 31, No. 2, pp. 115–130. DOI: 10.1177/0143034310361674.
22. **Vidal, J. (1977).** Real-time detection of brain events in EEG. Proceedings of the IEEE, Vol. 65, No. 5, pp. 633–641. DOI: 10.1109/proc.1977.10542.
23. **Jung, T., Makeig, S., Stensmo, M., Sejnowski, T. (1997).** Estimating alertness from the EEG power spectrum. IEEE Transactions on Biomedical Engineering, Vol. 44, No. 1, pp. 60–69. DOI: 10.1109/10.553713.
24. **Yaomanee, K., Pan-Ngum, S., Ayuthaya, P. I. N. (2012).** Brain signal detection methodology for attention training using minimal EEG channels. Tenth International Conference on ICT and Knowledge Engineering, pp. 84–89 DOI: 10.1109/ictke.2012.6408576.
25. **Belle, A., Hargraves, R. H., Najarian, K. (2012).** An automated optimal engagement and attention detection system using electrocardiogram. Computational and Mathematical Methods in Medicine, Vol. 2012, pp. 1–12. DOI: 10.1155/2012/528781.
26. **Li, Y., Li, X., Ratcliffe, M., Liu, L., Qi, Y., Liu, Q. (2011).** A real-time EEG-based BCI system for attention recognition in ubiquitous environment. Proceedings of International Workshop on Ubiquitous affective awareness and intelligent interaction, pp. 33–40. DOI: 10.1145/2030092.2030099.
27. **Calderón-Garcidueñas, L., Mora-Tiscareño, A., Ontiveros, E., Gómez-Garza, G., Barragán-Mejía, G., Broadway, J., Chapman, S., Valencia-Salazar, G., Jewells, V., Maronpot, R. R., Henríquez-Roldán, C., Pérez-Guillé, B., Torres-Jardón, R., Herrit, L., Brooks, D., Osnaya-Brizuela, N., Monroy, M. E., González-Maciel, A., Reynoso-Robles, R., Villarreal-Calderon, R., et al. (2008).** Air pollution, cognitive deficits and brain abnormalities: a pilot study with children and dogs. Brain and Cognition, Vol. 68, No. 2, pp. 117–127. DOI: 10.1016/j.bandc.2008.04.008.
28. **Wang, S., Zhang, J., Zeng, X., Zeng, Y., Wang, S., Chen, S. (2009).** Association of traffic-related air pollution with children's neurobehavioral functions in Quanzhou, China. Environmental Health Perspectives, Vol. 117, No. 10, pp. 1612–1618. DOI: 10.1289/ehp.0800023.
29. **Suglia, S. F., Gryparis, A., Wright, R. O., Schwartz, J., Wright, R. J. (2007).** Association of black carbon with cognition among children in a prospective birth cohort study. American Journal of Epidemiology, Vol. 167, No. 3, pp. 280–286. DOI: 10.1093/aje/kwm308.
30. **Pope, C. A., Dockery, D. W. (2006).** Health effects of fine particulate air pollution: lines that connect. Journal of the Air and Waste Management Association, Vol. 56, No. 6, pp. 709–742. DOI: 10.1080/10473289.2006.10464485.
31. **Russell, A. G., Brunekreef, B. (2009).** A focus on particulate matter and health. Environmental Science and Technology, Vol. 43, No. 13, pp. 4620–4625. DOI: 10.1021/es9005459.
32. **Zivin, J. G., Neidell, M. (2012).** The impact of pollution on worker productivity. American Economic Review, Vol. 102, No. 7, pp. 3652–3673. DOI: 10.1257/aer.102.7.3652.
33. **Miller, S., Vela, M. A. (2013).** The effects of air pollution on educational outcomes: evidence from Chile. IDB Working Paper, No. IDB-WP-468. DOI: 10.2139/ssrn.2370257.

34. Zaletelj, J., Košir, A. (2017). Predicting students' attention in the classroom from kinect facial and body features. EURASIP Journal on Image and Video Processing, Vol. 2017, No. 1. DOI: 10.1186/s13640-017-0228-8.

35. Usha-Devi, S., Balakrishnan, S. R., Nayak, S. N. K., Behera, S. D. (2016). Compendium of environment statistics, Odisha 2016. Directorate of Economics and Statistics. Computer Centre, Directorate of Economics and Statistics, Odisha, Bhubaneswar.

36. ENVIS (2023). Air Pollution. envis.org/index.php? option=com_content andview=categoryandid=18andItemid=159.

37. State Pollution Control Board (2013). Annual report 2018-2019. ospcboard.org/wp-content/plugins/publication//uploads/files_160 4314558_1246256032.pdf

38. Ministerio del Medio Ambiente (2012). Informe del estado del medio ambiente. 2nd Edition, Gobierno de Chile.

39. Ostro, B. D., Eskeland, G. S., Sanchez, J. M., Feyzioglu, T. (1999). Air pollution and health effects: a study of medical visits among children in Santiago, Chile. Environmental Health Perspectives, Vol. 107, No. 1, pp. 69–73. DOI: 10.1289/ehp.9910769.

40. Burnett, R., Dales, R., Raizenne, M., Krewski, D., Summers, P., Roberts, G., Raadyoung, M., Dann, T., Brook, J. (1994). Effects of low ambient levels of ozone and sulfates on the frequency of respiratory admissions to Ontario hospitals. Environmental Research, Vol. 65, No. 2, pp. 172–194. DOI: 10.1006/enrs.1994.1030.

41. Marques, G., Pitarma, R. (2019). A cost-effective air quality supervision solution for enhanced living environments through the internet of things. Electronics, Vol. 8, No. 2, pp. 170. DOI: 10.3390/electronics8020170.

42. Marques, G., Pitarma, R. (2019). A cost-effective air quality supervision solution for enhanced living environments through the internet of things. Electronics, Vol. 8, No. 2, pp. 170. DOI: 10.3390/electronics8020170.

43. Honeywell (2021). HPM series particulate matter sensors (DataSheet) sensing.honey well.com/honeywell-sensing-particulate-hpm-series-datasheet-32322550.pdf.

44. SHARP (2008). GP2Y1010AU0F data sheet. www.sparkfun.com/datasheets/Senso rs/gp2y1010au_e.pdf

45. Huawei Electronics (2020). MQ-7 gas sensor technical data. www.sparkfun.com/datas heets/Sensors/Biometric/MQ-7.pdf.

46. DDScientific (2023). GS+4NO2-nitrogen dioxide sensor (NO2) product data sheet. aqicn. org/air/ view/ sensor/ spec/no2. gs4 no 2.pdf.

47. China Total (2023). MQ136 semiconductor sensor for sulfur dioxide. www.china-total.com/ Product/ meter/ gassensor/ MQ 13 6.pdf.

48. Sensors Total (2023). MQ131 semiconductor sensor for ozone. www.sens orsportal.com/ DOWNLOADS/MQ131.pdf.

49. Raspberry Pi (2023). Raspberry Pi 3 model B. www.raspberrypi.com/products/raspberry-pi-3-model-b/.

50. Belle, A., Hargraves, R. H., Najarian, K. (2012). An automated optimal engagement and attention detection system using electrocardiogram. Computational and Mathematical Methods in Medicine, Vol. 2012, pp. 1–12. DOI: 10.1155/2012/528781.

51. Magosso, E., Crescenzio, F. D., Ricci, G., Piastra, S., Ursino, M. (2019). EEG alpha power is modulated by attentional changes during cognitive tasks and virtual reality immersion. Computational Intelligence and Neuroscience, Vol. 2019, pp. 1–18. DOI: 10.1155/2019/7051079.

52. Li, X., Zhao, Q., Hu, B., Liu, L., Peng, H., Qi, Y., Mao, C., Fang, Z., Liu, Q. (2010). Improve affective learning with EEG approach. Computing and Informatics, Vol. 29, pp. 557– 570.

53. Hasegawa, C., Oguri, K. (2006). The effects of specific musical stimuli on driver's drowsiness. Proceedings of the IEEE Intelligent Transportation Systems Conference, pp. 817–822. DOI: 10.1109/itsc. 2006.1706844.

54. NeuroSky (2021). EEG biosensors. neurosky. com/ biosensors/ eegsensor/ biose nsors/.

55. Aldayel, M., Ykhlef, M., Al-Nafjan, A. (2020). Deep learning for EEG-based preference classification in neuromarketing. Applied Sciences, Vol. 10, No. 4, pp. 1525. DOI: 10.33 90/app10041525.

56. Costantini, G., Casali, D., Todisco, M. (2010). An SVM based classification method for EEG signals. Proceedings of the International Conference on Circuits, pp. 107– 109.

57. Zaletelj, J., Košir, A. (2017). Predicting students' attention in the classroom from kinect facial and body features. EURASIP Journal on Image and Video Processing, Vol. 2017, No. 1. DOI: 10.1186/s13640-017-0228-8.

58. Liu, N., Chiang, C., Chu, H. (2013). Recognizing the degree of human attention using EEG signals from mobile sensors. Sensors, Vol. 13, No. 8, pp. 10273–10286. DOI: 10.3390/s130810273.

59. Ham, J. C., Zweig, J., Avol, E. L. (2011). Pollution, test scores and the distribution of academic achievement: evidence from California schools 2002-2008. Mothers and others for clean air. Mothersandothersforcleanair.org/research_art icles/pollution-test-scores-and-the- distribution -of-academic-achievement-evidence-from-cali fornia-schools-2002-2008/.

60. Xia, S., Huang, D., Jia, H., Zhao, Y., Li, N., Mao, M., Lin, H., Li, Y., He, W., Zhao, L. (2019). Relationship between atmospheric pollutants and risk of death caused by cardiovascular and respiratory diseases and malignant tumors in Shenyang, China, from 2013 to 2016: an ecological research. Chinese Medical Journal, Vol. 132, No. 19, pp. 2269–2277. DOI: 10.1097/cm9.0000000000000453.

Article received on 07/01/2024; accepted on 17/03/2024.
**Corresponding author is Bibudhendu Pati.*

ISSN 2007-9737

PAREX: A Novel exFAT Parser for File System Forensics

Gaurav Gogia*, Parag Rughani

National Forensic Sciences University,
India

gaurav-gogia@outlook.com, parag.rughani@gmail.com

Abstract. File systems, being one of the core components of any computational device, contain the most important information which makes them pivotal to any digital forensics investigation. However, file system parsing is a complex process. Existing file system forensic software are capable of processing large datasets but often at the cost of either performance or resource utilisation. Slow evidence processing has a direct impact on investigation time, while higher resource requirements have a monetary impact. Digital Forensics labs are often on a constrained budget in terms of both time and money. So, they often need to define priorities on a case-by-case basis. Another major concern for forensic investigators is correctness. Tools that suffer from memory management issues may generate inconsistent reports or worse yet, increase overall attack surface for malware that may pollute investigator's workstation. This research proposes a novel open-source exFAT file system parsing library. It has been validated against the current open-source state-of-the-art: The Sleuth Kit (TSK), on a dataset of disk images ranging from 1MiB to 1TiB. Experimental results indicate that the proposed tool is 40 times faster and 17 times more memory efficient than TSK.

Keywords. ExFat, parsers, file system forensics, digital forensics.

1 Introduction

File systems are blueprints that provide an arrangement for operating systems to efficiently & reliably store all the files. Modern file systems can scale up to billions of files with each file reaching sizes over multiple terabytes. Such features have made file systems the de facto structure of storing data in secondary storage.

Some of the most common file systems include NTFS, exFAT, EXT4, APFS, and XFS. As the default storage method, most of the data is stored in secondary storage. This effectively turns file systems into a gold-mine of artefacts for digital forensics investigators [28].

To extract these artefacts, numerous digital forensic tools have been developed [36]. Almost all modern digital forensic tools have a file system parsing module in them. A file system parser is a tool that understands the structure of the file system it has to read. The parser must understand all the pre-defined fields in the file system and read them while understanding their specific meanings.

Some parts of the file system contain meta-data while others contain contents of the files stored in them. A file system is similar to a JavaScript Object Notation (JSON) object or an Extensible Markup Language (XML) object in the sense that all three of them present different ways to organize data.

Almost all of the modern digital forensic tools have file system parsing capabilities. Some of these modern tools include TSK [41] & EnCase [32]. These forensics tools parse the file system to extract the artefacts and run analytics on top of them. However, file system parsing can be a complex process.

While digital forensics tools are able to handle terabytes of data, they are often slow to process the large volume of data [29]. Processing performance and resource utilisation are inversly related, high performance processing requires higher memory in most cases. For instance, parsing a file system with a large number of files and directories can result in high memory usage, which can slow down the parsing process.

Fig. 1. exFAT file system not detected at offset '0'

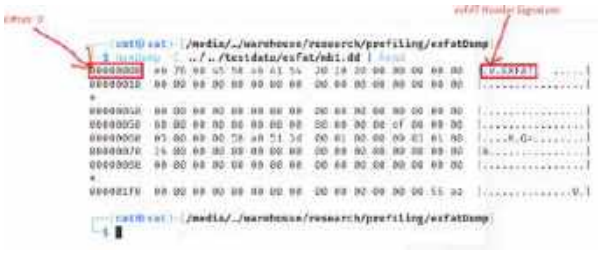


Fig. 2. exFAT file system header signature at offset '0'

When file system parsing is slow or inefficient, it slows down investigations. In certain instances, the indexing of an entire disk image with millions of files can be a time-consuming process, particularly when dealing with legacy equipment. The "Experiment" section of this research offers empirical evidence derived from testing various scenarios on five different platforms.

This research focuses on open-source file system parsers. Specifically designed for the exFAT file system [2], aiming to contribute an efficient and fast exFAT parser. The exFAT file system was selected for its compatibility with most of the major electronic devices [20, 23, 43, 39, 40, 1]. The "Related Work" section provides a more comprehensive discussion of this aspect.

1.1 Problem Statement

Forensic investigations often suffer from large backlog of cases due to slow, bulky, and inefficient digital forensic tools [29]. Digital evidence is relevant in about 90% of the cases [26]. The amount of data that must be acquired, analyzed, indexed, processed, triaged, and reported, keeps increasing every passing day. Storage devices have evolved to preserve multiple terabytes of data.

While advanced forensic tools and high-powered hardware tools are available, a lot of digital forensics labs don't have the budget for these advanced tools [17]. Building a digital forensics lab can easily cost over $100,000 [33], this cost is excluding yearly maintenance cost of the lab. Labs that rely on open-source tools find themselves lagging behind because existing tools are slow and memory in-efficient.

Next sub-section will explain the current state of open source exFAT file system parsers for digital forensic tasks. According to a joint report by Google and Microsoft, a staggering 70% of all security bugs can be traced back to memory safety issues [47, 48]. Although there are other open-source tools available for file system parsing, such as dfir_ntfs [30] & exfatDump [18] written in Python [37], they appear to have limited functionality. For instance, dfir_ntfs, is designed for parsing the Volume Boot Record (VBR), Master Boot Record (MBR) structures, and file metadata without the ability to read or extract any entries.

While, exfatDump, published in October 2015, unfortunately, seems to be non-operational. As shown in Figure 1, exfatDump fails to identify the start of the exFAT file system at the specified offset 0, which is indicated using hexdump [16] tool in figure 2. In contrast, the proposed solution of this research offers a robust and comprehensive approach to parsing file system structures.

1.2 Contribution

The primary contribution of this research is LIBXFAT, an exFAT file system parsing library [3]. Additionally, a secondary add-on to the contribution is PAREX, a Command Line Interface (CLI) tool built on top of LIBXFAT [4]. PAREX offers four options for parsing disk images, which are further explained in Table 1.

The main features of these software tools include listing all entries in the file system, including deleted and deleted entries, as well as extracting file contents in a forensically sound manner. Both LIBXFAT and PAREX were developed using the Go programming language [14], which was selected for its simplicity, easy concurrency, and built-in memory safety features.

Table 1. PAREX options

Parsing Option	Operation
0	List Root Entries
1	List All Entries - With Metadata
2	List All Entries - With Count
3	Extract All Entries - Collected
4	Extract All Entries - Recursively

Table 2. Indexed files per image

Image Size	Indexed Files
1 MiB	7
512 MiB	4540
1 GiB	11336
10 GiB	240295
32 GiB	442
64 GiB	313030

The memory safety aspect of the Go has been validated by Felix A. Wolf et al., as cited in reference [45]. To ensure the correctness of LIBXFAT, its results were compared with those obtained from TSK and Autopsy [5]. Furthermore, for benchmarking purposes, LIBXFAT was profiled against TSK using various parameters. Detailed explanations of these experiments can be found in the experiments section.

1.3 Outline

The rest of the paper is organized in 5 major sections: Section 2 ”Related Work” discusses some related research work with the current state of file system parsing tools. Section 3 ”Background” explains the structure of exFAT file system. This section can be skipped by those readers who understand this file system well.

Section 4 “Experiment” presents experiments, experimental methodology, and the data generated during those experiments. In Section 5 “Discussion”, experimental results are interpreted; Section 6 “Conclusion” concludes and discusses future research work and potential applications of this work.

2 Related Work

This section reviews studies on file system forensics, highlights the importance of exFAT from a forensic perspective, and explores existing software solutions. It focuses on solutions that offer API/Library integration for developers to promote inter-operability in the field.

File System Forensics: The significance of file system forensics is paramount; virtually all digital artefacts can be traced back to the file system. A recent paper explored the application of machine learning algorithms for identifying contraband within file systems [28]. Extensive research has been conducted regarding forensic and anti-forensic techniques for various file systems.

One such study proposed a novel file recovery algorithm designed to recover deleted files from the FAT32 file system [8]. Another research introduced a scheme aimed at detecting data in FAT32 file systems that leaves no traces [46]. However, investigating file systems isn't solely limited to recovery files or identifying data streams. For instance, ExtSFR is a scalable file recovery framework that is compatible with EXT file systems [19].

APFS, another crucial file system, has been sparsely studied due to its proprietary, closed-source nature. Researchers interested in developing file recovery algorithms for this file system had to resort to reverse engineering [36]. Similarly, the proprietary Resilient File System (ReFS) also necessitates reverse engineering to extract valuable information [32].

These efforts pose various legal and technical questions around the process of reverse engineering a file system [42]. However, reverse engineering and file recovery are not the sole use cases in file system forensics. Another scenario involves the simple reading and classification of files.

This has spurred research into classification algorithms, with some based on neural networks that traverse the file system to identify contraband [27]. File system forensics has indeed spurred a numerous research initiatives. To that end, this paper will primarily focus on the exFAT file system.

Table 3. exFAT region layout with offset & size in sectors

Sub-Region Name	Offset (Hex)	Size (Decimal)
Main Boot Region		
Main Boot Sector	0x0	1
Main Extended Boot Sectors	0x1	8
Main OEM Parameters	0x9	1
Main Reserved	0xA	1
Main Boot Checksum	0xB	1
Backup Boot Region		
Backup Boot Sector	0xC	1
Backup Extended Boot Sectors	0xD	8
Backup OEM Parameters	0x15	1
Backup Reserved	0x16	1
Backup Boot Checksum	0x17	1
FAT Region		
FAT Alignment	0x18	FatOffset – 24
First Fat	FatOffset	FatLen
Second Fat	FatOffset + FatLen	FatLen * (FatCount – 1)
Data Region		
Cluster Heap Alignment	FatOffset + FatLen * FatCount	ClusterHeapOffset – (FatOffset + FatLen * FatCount)
Cluster Heap	ClusterHeapOffset	ClusterCount * 2
Excess Space	ClusterHeapOffset + ClusterCount*2	VolumeLen – (ClusterHeapOffset + ClusterCount * 2)

exFAT Forensics: The exFAT file system has emerged as the de facto standard for removable storage devices and those utilizing NAND flash storage technology, including thumb drives, SDXC cards, eMMC storage in laptops, and more.

A key factor driving its widespread adoption is the deliberate design choice made by its creators—minimising write operations to promote the longevity of storage devices. Another pivotal aspect is its compatibility with major operating systems such as Windows, MacOS, Ubuntu, Android, and other Unix-based systems [15]. Forensic workstations leverage exFAT file systems to ensure seamless interoperability across different operating systems [20].

Despite its popularity, the algorithms used to parse and analyse devices employing the exFAT system are not publicly available, posing a challenge to comprehensive forensic analysis [23]. The significance of the exFAT file system is underscored by Yves Vandermeer et al., who conducted an in-depth study on its data structure [43]. Furthermore, various studies highlight the use of exFAT file systems in a diverse array of devices, from medical equipment to drones [39, 40, 1]. Consequently, advancing our understanding and capabilities in exFAT forensics holds paramount importance.

Open Source Digital Forensics: Several digital forensics software programs capable of parsing the exFAT file system currently exist.

Table 4. Number of entires per disk image

Size	All	Root	Indexed	Deleted
1 MiB	10	6	10	0
512 MiB	4734	19	4733	1
1 GiB	11580	20	11561	19
5 GiB	1	9	9	0
10 GiB	246910	13	246845	65
25 GiB	11	11	11	0
32 GiB	556	13	556	0
40 GiB	20	20	20	0
64 GiB	325517	112	324865	652
128 GiB	533292	150	532444	848
256 GiB	1375122	152	1374820	302
500 GiB	45	45	45	0
512 GiB	2550344	152	2550212	132
1 TiB	5057669	1546	5053570	4099

However, the process of validating, benchmarking, and developing trust in these tools presents a considerable challenge [7]. Although open-source file system forensics software are available [41, 40], their over-reliance on memory-unsafe programming languages such as C & C++ can cause supply chain security challenges.

Multiple entities, including Google and Microsoft, along with independent researchers, have reported that over 70% of memory-related flaws in operating systems and browsers can be directly attributed to code written in C or C++ [47, 48, 49, 12].

These memory-related bugs, if exploited, can compromise a digital forensics workstation causing damage to all the cases being investigated on the same machine or on the same local network, depending on the extent of the exploit.

The Go programming language, developed by Google, offers a safer alternative [45]. Its design goals are simplicity, ease of development, and high performance. When compared to other memory-safe programming languages, the performance of Go is comparable to C++ [35, 11, 22, 34].

However, only one open-source library has been identified to date [10] that is written in Go programming language, but it lacks features such as recursive file system traversal and access to metadata information like the number of clusters a file contains.

This paper introduces a new open-source library and a CLI tool for parsing the exFAT file system. This library has been validated with and benchmarked against industry standard tools. Validation and benchmark tests are explained in more detail in Section 4 ”Experiment” section.

3 Background

The exFAT file system was developed by Microsoft to address the limitations of the FAT32 file system, particularly in relation to flash storage devices like SD cards. The exFAT file system comprises three primary regions: the Boot Region (also known as the superblock), the FAT Region, and the Data Region. The Boot Region, occupying the first 512 bytes, contains crucial metadata and initialization data, including the file system signature and parameters.

The FAT Region, as the name suggests, stores the file allocation table, which manages file and directory locations. The Data Region stores the actual contents and metadata of all the files and folders. A comprehensive study by Julian Heeger et al. provides detailed insights into the architecture and functioning of the exFAT file system [15]. Table 3 explains subsections of these regions with their offsets [2].

4 Experiment

To evaluate the correctness and performance for PAREX software (powered by LIBXFAT), functional and benchmarking experiments were carried out. For benchmarking experiments FLS software (powered by TSK) was used as the control/standard software against all the comparison was made. The upcoming subsections will provide detailed explanations of the experiment setup, the experiment itself, and the corresponding results.

Fig. 3. Verify metadata - PAREX

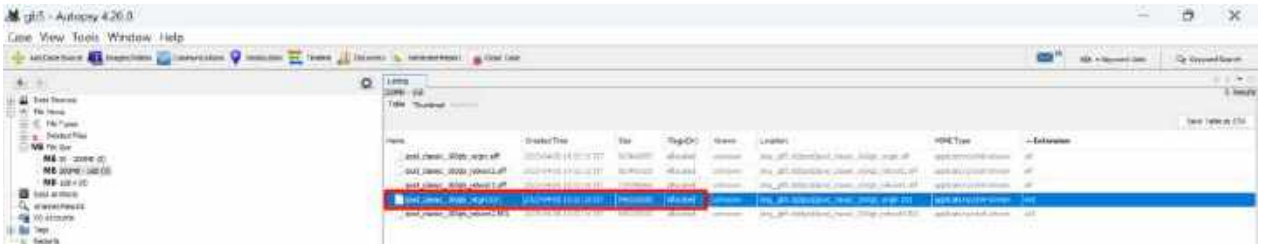


Fig. 4. Verify metadata - autopsy

4.1 Experiment Setup

To validate both PAREX and FLS, the first step involved generating a dataset of raw disk images. Most of the disk images were generated synthetically by creating several files and folders in an external storage device. Additionally, a portion of the dataset was downloaded from NIST CFReDS repositories [31].

The downloaded files were saved into a separate partition in a Virtual Machine. Raw disk images were created out of the partitions where the files were downloaded. Disk images ranging from 1MiB to 1TiB were created to cover a range of scenarios. Experiment setup is divided into two parts: Functional Tests Setup & Benchmark Tests Setup.

They explain the experiment environment and profiling methods used to conduct the experiment. Within the context of this study, the terms 'entry' & 'entries' are employed to denote an 'entry' & a number of 'entries' within the file system. This term is used universally to represent both files and folders. Table 4 exhibits the total number of entries, root entries, indexed entries, and deleted entries for a disk image, along with their respective sizes.

4.1.1 Functional Test Setup

The experimental environment for the functional tests was intentionally simple. A single platform was chosen to evaluate the basic functionality of PAREX software developed within this research. Performance assessments of all the tools were executed using GNU/Time software [13] within a Windows Subsystem for Linux 2 (WSL-2) environment [6].

4.1.2 Benchmark Test Setup

The benchmark tests were performed in five different environments: WSL-2 [6], Anarchy Linux [9], Windows 10 Professional [24], Windows 11 Professional [25], and Kali Linux [21]. These environments were created using VMWare Workstation Pro 17 [44].

Windows 10, Windows 11, Anarchy Linux, & Kali Linux environments were created with 4 vCPUs and 6 GiB RAM. A 12 TiB 7200RPM external HDD was used for storage. While on the other hand, WSL-2 had 16 vCPUs, 32 GiB RAM, and 1 TiB internal M2 SSD storage drive.

Algorithm 1 Process Profiling Algorithm

start_time ← current time
p ← Process object for proc.pid
Initialise empty lists: cpu_use_list, mem_use_list, thread_count_list, read_bytes_list, write_bytes_list
while proc is running **do**
    cpu ← CPU percent of p divided by total number of CPUs
    Append cpu to cpu_use_list
    mem ← memory used by p
    Append mem to mem_use_list
    Append number of threads used by p to thread_count_list
    Get I/O counters for p
    Append read bytes and written bytes to read_bytes_list and write_bytes_list respectively
    Sleep for sampling_rate seconds
end while
end_time ← current time
avg_cpu_use ← Average of cpu_use_list
avg_ram_use ← Average of mem_use_list
avg_thread_count ← Average of thread_count_list
avg_io_read ← Average of read_bytes_list
avg_io_write ← Average of write_bytes_list
return avg_cpu_use, avg_ram_use, avg_thread_count, avg_io_read, avg_io_write

To profile all these experiments, a python [37] script was written using psutil [38] library. This library helps in profiling system events like memory use, execution time, thread count, processor use, disk reads/writes etc.

The version of the library at the time of writing this paper is v5.9.5. This library has a proven record and is being actively maintained on GitHub by many contributors. Algorithm 1 explains the profiling script.

4.2 Functional Tests

For the functional tests, all the files were extracted out of the disk images using PAREX to validate the correctness of the CLI tool. Table 2 shows image size and the number of indexed files per disk image.

To verify the correctness of PAREX, results of data parsing and extraction were compared with the results of TSK & Autopsy. Please note that an additional command in PAREX was executed for this experiment to list out all the files and their metadata out of the disk image.

Figures 3 & 4 represent matching metadata between PAREX & Autopsy to solidify the correctness of exFAT file system parsing in PAREX. On the other hand, figure 7 represents matching SHA-256 hash of the extracted files by PAREX & TSK illustrating correctness of file extracted by PAREX.

Mean and Standard Deviation out of experiment data was calculated to compare performance, efficiency, and consistency between PAREX and FLS. Upcoming sub-sections will delve into the details of the experiemnts and present results.

4.2.1 List Root Entries

In the first benchmark test, FLS was executed in its default state without any flags, while PAREX(powered by LIBXFAT) was run with the 0 option. This option parses root dir entry and returns them for the user, please find table 1 for more details. This approach ensured that both tools used a 0 offset of the disk image as the starting point and exclusively returned root entries from the disk image.

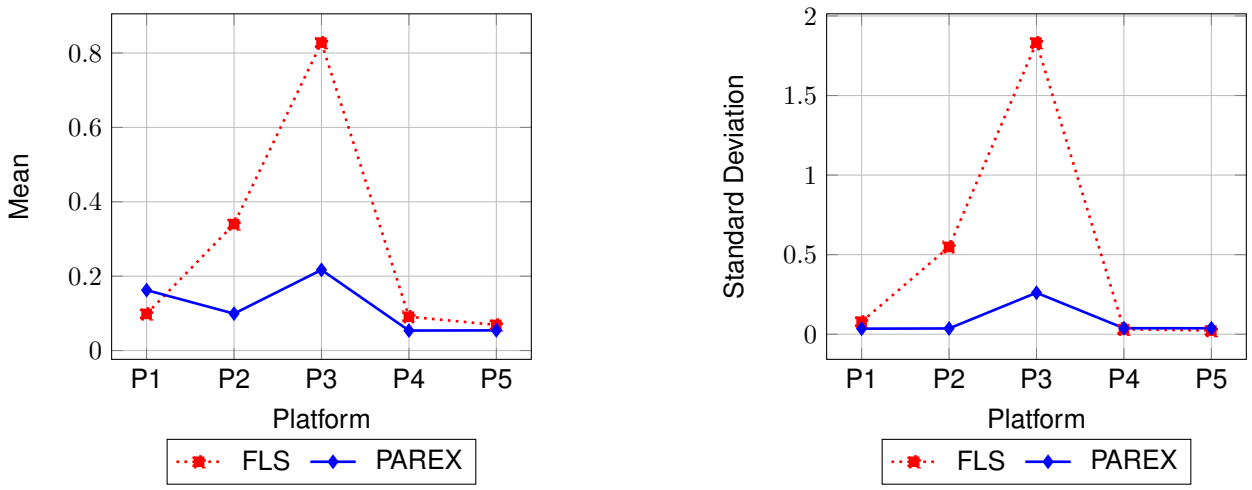


Fig. 5. List root entries — execution time (seconds) — lower is better

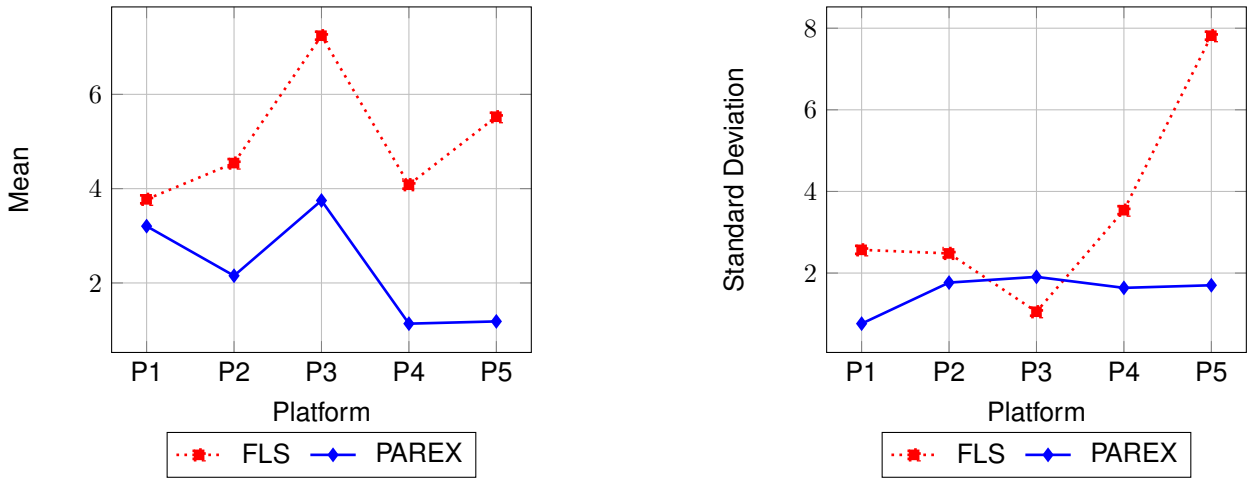


Fig. 6. List root entries — RAM use (MB) — lower is better

4.2.2 List All Entries

In the second benchmark test, FLS was run with options '-u -r' while PAREX was run with option '2'. This option lists out all the indexed entries in the file system while keeping track of count of number of entries, please find table 1 for more details. This approach ensured that both tools used a 0 offset of the disk image as the starting point and exclusively returned all the indexed endtries from the disk image.

4.3 Experiment Results

Mean & Standard Deviation was calculated for all the statistical data that was acquired by profiling the experiments run to benchmark both PAREX and FLS software tools. Results of these experiments have been visualized to clearly state the difference in performance, efficiency, and consistency between the two tools. Figures 5, 6, 8, and 9 represent comparative analysis between PAREX and FLS on various platforms through mean execution time, standard

Fig. 7. Verify file extraction

deviation in execution time, mean RAM use, and standard deviation in RAM use. The following list elucidates the platforms on which these experiments were conducted:

– P1 - WSL 2,
– P2 - Windows 10,
– P3 - Windows 11,
– P3 - Arch Linux,
– P4 - Kali Linux.

4.4 Caveats

This study has several important caveats that must be considered when interpreting its findings. Prrofiling in a Windows Subsystem for Linux 2 (WSL-2) environment is currently not fully reliable. The high level of abstraction that WSL-2 introduces an inherent challenge in accurately profiling all parameters.

This limitation potentially impacts the precision and consistency of our results obtained from this environment. To alleviate this concern, identical experiments were conducted on Anarchy Linux, Kali Linux, Windows 10, and Windows 11 virtual machines were conducted.

The diverse range of environments helps in establishing a comprehensive and more reliable picture of the software's execution time. Secondly, the accuracy of profiled data is inversely proportional to the sampling rate.

As the sampling rate decreases, the chances of obtaining accurate profiling data diminishes. This phenomenon occurs because lower sampling rates have a reduced ability to capture all system state changes accurately. To examine this effect and capture data at different levels of granularity, we performed profiling at sampling rates of 0ms, 50ms, and 100ms.

Lastly, the accuracy of profiled data can also be affected if an operation runs faster than the sampling rate. This discrepancy can lead to the operation being entirely missed by the profiler, especially for those operations that complete within a time frame shorter than the sampling rate. This occurrence introduces another potential source of error in the profiling data.

Therefore, it's critical to understand that the results of this study are subjected to these inherent limitations of the profiling process. Future work could focus on developing methods to mitigate these issues and enhance the accuracy of profiling data.

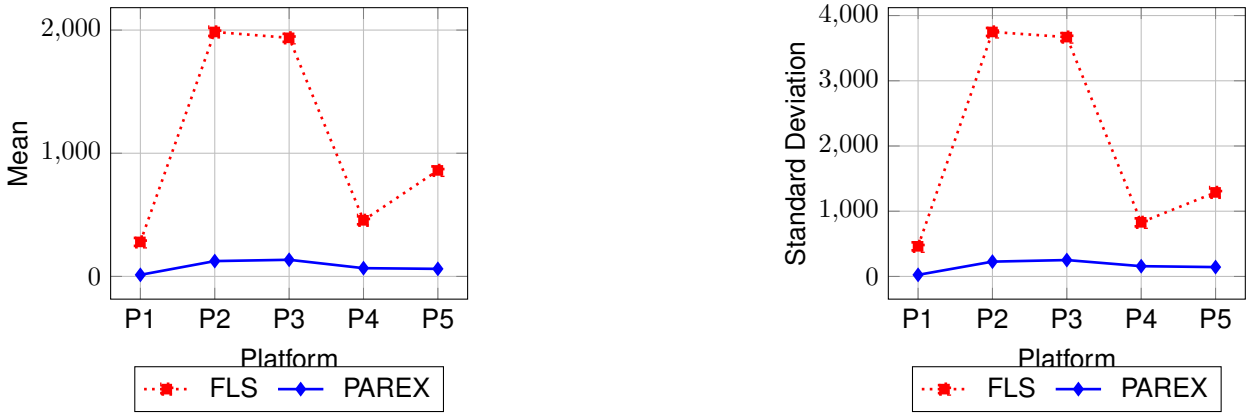


Fig. 8. List all entries — execution time (seconds) — lower is better

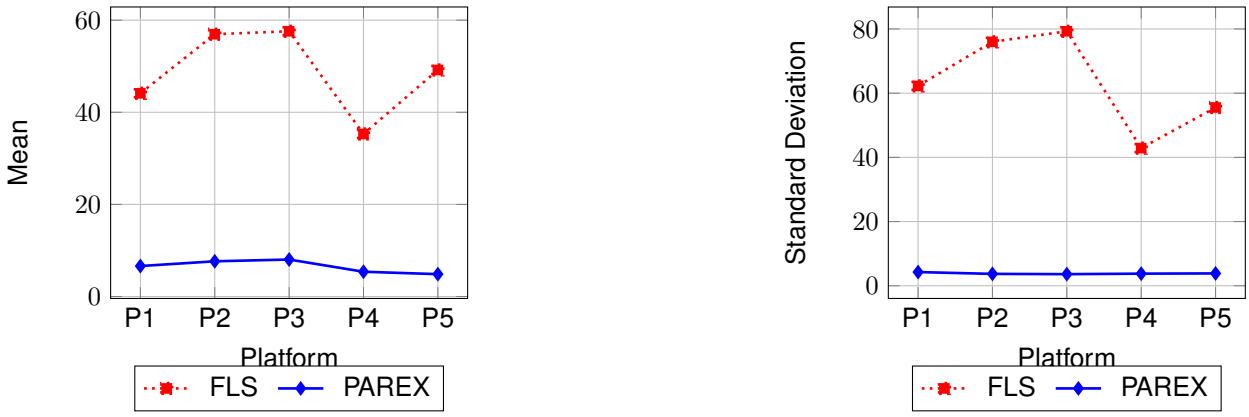


Fig. 9. List all entries — RAM use (MB) — lower is better

5 Discussion

This section presents the outcomes of functional and benchmark tests.

The experimental results indicate that the PAREX and LIBXFAT library accurately parses the exFAT file system, effectively identifying root entries, traversing all directories and sub-directories to locate the remaining entries, and extracting file content in a forensically sound manner.

The benchmark tests reveal that the PAREX, is significantly faster and more memory efficient than FLS. Experiments also reveal that PAREX software performs much more consistently across different platforms.

This is another important finding, as it means that PAREX can be used to process large exFAT formatted devices reliably with high speed while keeping minimum memory footprint on the investigator's workstation. However, this research does have a limitation: the absence of active detection for deleted entries.

While the PAREX can identify obviously deleted or deleted entries evidenced by 0x0 listed as the entry cluster offset, it does not carry out any advanced operations to detect less obvious deleted entries.

Furthermore, PAREX does not employ any statistical or pattern matching techniques to identify deleted directory entries, it can only detect deleted file entries.

These features can be added in future to arm PAREX with more features for forensic artefact analyses. Overall, the findings of the research are positive. PAREX is a promising tool for recovering data from exFAT formatted devices. However, future work should focus on addressing the limitation of the research by developing methods to actively detect deleted entries.

6 Conclusion

In this study, an open-source library and a CLI tool were developed for parsing the exFAT file system. To validate correctness and performance, deep profiling and benchmarking tests were conducted using the psutil library on five different platforms: WSL-2, Anarchy Linux, Kali Linux, Windows 10, & Windows 11. The developed tools were benchmarked against industry standard open-source tools: The Sleuth Kit (TSK) & Autopsy. The results demonstrate that the developed tools are over 40 times faster than the control set while also being 17 times more memory efficient.

The developed software consistently present effective and efficient results over multiple platforms. These results directly impact the cost of acquisition and maintenance of workstations and other associated computer hardware, be it on-premises or on cloud. However, to further enhance the software, several optimization strategies can be implemented.

These include improving the handling of multiple goroutines, implementing thread-pooling for larger objects, and conducting deeper profiling tests to identify and eliminate unnecessary object allocations and deallocations. Moreover, future research prospects involve addressing the limitation of active deleted file detection and deleted file recovery by developing additional features.

References

1. **Allen-Barton, T. E., Bin-Azhar, M. A. H. (2018).** Open source forensics for a multi-platform drone system. Computing, Digital Forensics and Cybersecurity, pp. 83–96. DOI: 10.1007/978-3-319-73697-6_6.
2. **alvinashcraft (2022).** Exfat file system specification - win32 apps. https://learn.microsoft.com/en-us/windows/win32/fileio/exfat-specification.
3. **aoiflux (2023).** Libxfat. https://github.com/aoiflux/libxfat.
4. **aoiflux (2023).** Parex. https://github.com/aoiflux/parex.
5. **Autopsy (2023).** Digital forensics. https://www.autopsy.com/.
6. **Barnes, H. (2021).** Pro Windows subsystem for Linux (WSL). Apress. DOI: 10.1007/978-1-4842-6873-5.
7. **Bhat, W. A., AlZahrani, A., Wani, M. A. (2021).** Can computer forensic tools be trusted in digital investigations?. Science & Justice, Vol. 61, No. 2, pp. 198–203. DOI: 10.1016/j.scijus.2020.10.002.
8. **Chen, B., Guan, J., Wang, H., Yao, G. (2021).** A novel data recovery algorithm for FAT32 file system. 2021 2nd International Conference on Information Science and Education (ICISE-IE), pp. 605–608. DOI: 10.1109/ICISE-IE53922.2021.00143.
9. **Community, A. L. (2023).** Anarchy Installer. https://anarchyinstaller.gitlab.io/.
10. **dsoprea (2022).** go-exfat. https://github.com/dsoprea/go-exfat.
11. **Fua, P., Lis, K. (2020).** Comparing python, go, and C++ on the n-queens problem. http://arxiv.org/abs/2001.02491. DOI: 10.48550/arXiv.2001.02491.
12. **Gao, Y., Chen, L., Shi, G., Zhang, F. (2018).** A comprehensive detection of memory corruption vulnerabilities for C/C++ programs. 2018 IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications

(ISPA/IUCC/BDCloud/SocialCom/SustainCom), pp. 354–360. DOI: 10.1109/BDCloud.2018.00062.

13. **GNU (2023).** GNU time - GNU project - free software foundation. https://www.gnu.org/software/time/.
14. **Google (2023).** The go programming language. https://go.dev/.
15. **Heeger, J., Yannikos, Y., Steinebach, M. (2022).** An introduction to the exFAT file system and how to hide data within. Journal of Cyber Security and Mobility, Vol. 11, No. 2, pp. 239–264. DOI: 10.13052/jcsm2245-1439.1125.
16. **Kerrisk, M. (2023).** hexdump(1) - Linux manual page. https://www.man7.org/linux/man-pages/man1/hexdump.1.html.
17. **Lawrence, T., Karabiyik, U., Shashidhar, N. (2018).** Equipping a digital forensic lab on a budget. 2018 6th International Symposium on Digital Forensic and Security (ISDFS), pp. 1–7. DOI: 10.1109/ISDFS.2018.8355345.
18. **Iclevy (2023).** An experimental tool for forensic analysis of ExFAT filesystem. https://github.com/lclevy/exfatDump.
19. **Lee, S., Jo, W., Eo, S., Shon, T. (2020).** ExtSFR: Scalable file recovery framework based on an Ext file system. Multimedia Tools and Applications, Vol. 79, No. 23, pp. 16093–16111. DOI: 10.1007/s11042-019-7199-y.
20. **Lin, X. (2018).** Building a forensics workstation. Introductory Computer Forensics: A Hands-on Practical Approach, pp. 53–89.
21. **Linux, K. (2024).** Penetration testing and ethical hacking Linux distribution. https://www.kali.org/.
22. **Lion, D., Chiu, A., Stumm, M., Yuan, D. (2022).** Investigating managed language runtime performance: why javascript and python are 8x and 29x slower than c++, yet java and go can be faster? Usenix ATC'22, pp. 835–852.
23. **Mason, S., Seng, D. (2017).** Electronic evidence. University of London Press. DOI: 10.14296/517.9781911507079.
24. **Microsoft (2023).** Download Windows 10. https://www.microsoft.com/en-in/software-download/windows10.
25. **Microsoft (2023).** Download Windows 11. https://www.microsoft.com/en-in/software-download/windows11.
26. **Miller, C. M. (2022).** A survey of prosecutors and investigators using digital evidence: A starting point. Forensic Science International: Synergy, Vol. 6, pp. 100296. DOI: 10.1016/j.fsisyn.2022.100296.
27. **Mohammad, R. M. (2018).** A neural network based digital forensics classification. 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA), pp. 1–7. DOI: 10.1109/AICCSA.2018.8612868.
28. **Mohammad, R. M. A., Alqahtani, M. (2019).** A comparison of machine learning techniques for file system forensics analysis. Journal of Information Security and Applications, Vol. 46, pp. 53–61. DOI: 10.1016/j.jisa.2019.02.009.
29. **Montasari, R., Hill, R. (2019).** Next-generation digital forensics: Challenges and future paradigms. 2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3), pp. 205–212. DOI: 10.1109/ICGS3.2019.8688020.
30. **msuhanov (2023).** An NTFS/FAT parser for digital forensics & incident response. https://github.com/msuhanov/dfir_ntfs.
31. **NIST (2023).** CFReDS Portal. https://cfreds.nist.gov/.
32. **Nordvik, R., Georges, H., Toolan, F., Axelsson, S. (2019).** Reverse engineering of ReFS. Digital Investigation, Vol. 30, pp. 127–147. DOI: 10.1016/j.diin.2019.07.004.
33. **Opolskii, V., Stupina, M. (2021).** Consumer-grade storage comparative

analysis in the context of digitalization of the agro-industrial complex. IOP Conference Series: Earth and Environmental Science, Vol. 937, No. 3, pp. 032079. DOI: 10.1088/1755-1315/937/3/032079.

34. **Pascal, C., Hurt, I., Mattson, T. G. (2022).** Towards a graphblas implementation for go. 2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 01–04. DOI: 10.1109/IPDPSW55747.2022.00052.
35. **Pascal, C., Herzeel, C., Verachtert, W. (2019).** Comparing ease of programming in C++, go, and java for implementing a next-generation sequencing tool. Evolutionary Bioinformatics, Vol. 15. DOI: 10.1177/1176934319869015.
36. **Plum, J., Dewald, A. (2018).** Forensic APFS file recovery. Proceedings of the 13th International Conference on Availability, Reliability and Security, pp. 1–10. DOI: 10.1145/3230833.3232808.
37. **Python (2023).** Welcome to Python.org. https://www.python.org/.
38. **Rodola, G. (2023).** Psutil: Cross-platform lib for process and system monitoring in Python. https://github.com/giampaolo/psutil.
39. **Schmitt, V. (2022).** Medical device forensics. IEEE Security & Privacy, Vol. 20, No. 1, pp. 96–100. DOI: 10.1109/MSEC.2021.3127490.
40. **Senturk, S., Apaydın, T., Yasar, H. (2020).** Image and file system support framework for a digital mobile forensics software. 2020 Turkish National Software Engineering Symposium (UYMS), pp. 1–3. DOI: 10.1109/UYMS50627.2020.9247055.
41. **sleuth kit, T. (2022).** The sleuth kit (Tsk) & autopsy: Open source digital forensics tools. https://sleuthkit.org/.
42. **Stoykova, R., Nordvik, R., Ahmed, M., Franke, K., Axelsson, S., Toolan, F. (2022).** Legal and technical questions of file system reverse engineering. Computer Law & Security Review, Vol. 46, pp. 105725. DOI: 10.1016/j.clsr.2022.105725.
43. **Vandermeer, Y., Le-Khac, N. A., Carthy, J., Kechadi, T. (2018).** Forensic analysis of the exFAT artefacts. DOI: 10.48550/arXiv.1804.08653.
44. **VMWare (2023).** Download VMware Workstation Pro. https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html.
45. **Wolf, F. A., Arquint, L., Clochard, M., Oortwijn, W., Pereira, J. C., Müller, P. (2021).** Gobra: Modular specification and verification of go programs. Computer Aided Verification, pp. 367–379. DOI: 10.1007/978-3-030-81685-8_17.
46. **Xu, S., Liu, F., Meng, L., Wang, L., Chang, X., Yang, W. (2022).** A scheme of traceless file deletion for windows FAT32 file system. Proceedings of the 2021 ACM International Conference on Intelligent Computing and its Emerging Applications, pp. 89–93. DOI: 10.1145/3491396.3506515.
47. **ZDNET (2022).** Chrome: 70% of all security bugs are memory safety issues. www.zdnet.com/article/chrome-70-of-all-security-bugs-are-memory-safety-issues/.
48. **ZDNET (2022).** Microsoft: 70 percent of all security bugs are memory safety issues. www.zdnet.com/article/microsoft-70-percent-of-all-security-bugs-are-memory-safety-issues/.
49. **Zhang, H., Wang, S., Li, H., Chen, T. H., Hassan, A. E. (2022).** A study of C/C++ code weaknesses on stack overflow. IEEE Transactions on Software Engineering, Vol. 48, No. 7, pp. 2359–2375. DOI: 10.1109/TSE.2021.3058985.

Article received on 12/01/2024; accepted on 11/03/2024.
**Corresponding author is Gaurav Gogia.*

ISSN 2007-9737

Artificial Intelligence in Latin American Universities: Emerging Challenges

Marina Fernández-Miranda[1,*], Daniel Román-Acosta[2], Adolfo A. Jurado-Rosas[3], Dolores Limón-Dominguez[4], Cristóbal Torres-Fernández[4]

[1] Universidad Tecnológica del Perú, Piura, Peru

[2] Universidad de Zulia, Maracaibo, Venezuela

[3] Universidad Privada Antenor Orrego, Piura, Peru

[4] Universidad de Sevilla, España

ajurador1@upao.edu.pe

Abstract. The integration of artificial intelligence in Latin American universities has raised ethical challenges among faculty members. Understanding and addressing these challenges is crucial for a successful implementation of artificial intelligence in the educational context. This study was conducted using a descriptive-explanatory quantitative approach, incorporating the opinions of 665 university professors from Latin America. Data was collected through surveys, with a Cronbach's alpha coefficient of 0.91. Analysis of the data revealed a range of ethical concerns among the educators regarding the utilization of artificial intelligence in education. These concerns vary in magnitude and nature, but they reflect a clear need to address and understand the ethical implications of artificial intelligence in the educational sphere. The adoption of artificial intelligence in Latin American higher education has raised ethical concerns. These concerns, while valid, should not be insurmountable barriers but points of reflection to optimize the integration of artificial intelligence in education. The study provides an essential overview for institutions, educators, and developers looking to implement AI in higher education, emphasizing the urgency of addressing these ethical challenges in an anticipatory and strategic manner.

Keywords. Artificial intelligence, ethical challenges, higher education, teacher perceptions, educational technology, pedagogical innovation.

1 Introduction

The constant emergence and evolution of technology in our daily lives have shaped a new reality teeming with both opportunities and challenges. Within this technological landscape, artificial intelligence (AI) has established itself as one of the most significant and transformative advancements of the 21st century.

AI has emerged as a transformative force in countless sectors, redefining the way we interact with technology and ourselves. Higher education, recognized as an essential cornerstone in the development of future leaders and professionals, has not been immune to this phenomenon, especially in the Latin American region [26].

Advanced tools, such as ChatGPT, appear to be the promising future of education [29-28]. However, their integration prompts important inquiries concerning ethics, discrimination, and privacy [25].

While there is positive evidence, such as the improvement in student motivation and performance through AIBO robots [3], the implementation of AI varies from one country to

another, with Argentina and Brazil leading the forefront [5].

The recent COVID-19 pandemic has catalyzed virtual teaching, enabling the use of AI to assess students' emotional states in real-time and make pedagogical adjustments [21-34]. Additionally, research focused on specific domains, including law [46-48], and advanced techniques like Deep Learning, have showcased AI's ability to predict academic success [49].

However, with these advancements, concerns about ethical responsibility [64], security [18], and gender biases [2] arise. Taheri and Aguayo [58] also propose a reevaluation of educational paradigms. Vivanco et al. [67] highlight innovations in AI, while Villamor [66] and Rosano and Corona [54] call for accountability in automated decision making.

In this ever-evolving landscape, a significant gap remains in understanding the ethical challenges of AI from the perspective of university faculty. Gómez [26] and Leal [35] in their study seek to fill that void, exploring and unraveling the ethical implications associated with the use of tools like ChatGPT in higher education.

Beyond investigating its application in specific areas such as journalism, mathematics, and other professional fields, this research endeavors to furnish a framework for an ethical and effective implementation of AI in higher education. Higher education serves as a pivotal force in shaping future leaders and professionals.

In this scenario, the integration of AI promises to revolutionize teaching, personalize learning, and enhance administrative efficiency. However, along with these advantages, significant ethical challenges arise, including concerns about privacy, bias, and discrimination.

It is imperative to understand and address these challenges during the integration of AI into the educational domain. By doing so, we not only optimize the educational experience with advanced technology but also ensure that it is carried out in alignment with our ethical and social values.

Therefore, it is paramount to scrutinize AI in education, especially tools like ChatGPT, from an ethical and pedagogical perspective.

1.1 Artificial Intelligence in Higher Education

The Emergence of Artificial Intelligence (AI) has redefined the contemporary educational landscape. Amid the COVID-19 pandemic, the relevance of AI in the educational sphere solidified, assisting and enhancing learning in challenging contexts [56]. Notably, emerging technological tools such as Augmented Reality and Virtual Reality have transcended classrooms, finding relevance in cultural spaces like museums, offering a new dimension to the interpretation and understanding of heritage [13].

In language teaching and tutoring, natural language processing tools like ChatGPT are emerging as pioneers, expanding the spectrum of pedagogical possibilities [14]. González [27] introduces the concept of neurodidactics, a field that bridges neuroscience and education. This convergence underscores how AI, through its capacity to interpret and adapt to individual needs, has the potential to democratize and personalize the learning experience.

However, with great advances come great responsibilities. UNESCO [62], while acknowledging the influence of AI, emphasizes the importance of personalized and efficient learning but also warns about inherent challenges such as biases and ethical implementation. Zhai [70] expands on this argument, asserting that tools like ChatGPT, though promising, require high-quality materials and well-prepared educators.

García et al. [24] corroborate this perspective by noting that the quality of AI-backed virtual education can match or even surpass traditional education. Finally, in the university sector, AI has emerged as a versatile and pervasive tool. For example, in journalism, it is being used to train students in crucial skills [26].

Modern pedagogical techniques, such as Problem-Based Learning (PBL) and gamification, have been revolutionized with the inclusion of AI, demonstrating the adaptability and versatility of this technology in various educational fields [45, 35, 69].

1.2 ChatGPT in Higher Education

ChatGPT, a language model developed by OpenAI and made available as open access, was launched

Fig. 1. Word cloud depicting the relevance of ChatGPT in university teaching

on November 30, 2022 [17]. This model is based on the GPT-3.5 architecture, with 'Chat' representing conversation and 'GPT' standing for Generative Pre-trained Transformer, highlighting its ability to generate coherent and contextually relevant text.

The primary purpose of Artificial Intelligence is to enhance human understanding and elevate the intellectual capacity of machines, with the aim of achieving maximum benefit [59].

The impact of ChatGPT in higher education is evident, as demonstrated by a study conducted by Gao, which showed that ChatGPT has the ability to generate plagiarism-free research summaries, although human reviewers correctly identified only 68% of them, posing challenges in the research review process in an AI-driven environment [38]. According to ChatGPT [70], it presents itself as a deep learning-based AI capable of generating text similar to humans.

Thanks to its training on a wide variety of text data, it can understand and respond to various natural language inputs, in addition to being customizable for specific tasks such as answering questions, language translation, text generation, and many other activities.

However, controversy surrounding ChatGPT centers on its potential use to generate text without requiring the necessary human effort, which could jeopardize students' acquisition of intellectual skills [23]. The launch of ChatGPT has the potential to impact education and research significantly.

Despite the opportunities AI offers to transform teaching and assessment, it also raises concerns about dependency instead of learning, necessitating a balance between automation and the promotion of creativity and critical thinking [1]. In this context, the perceptions of teachers and students regarding the impact of ChatGPT are being explored.

1.3 Tools and Applications of Artificial Intelligence in Education

Artificial Intelligence has emerged as a revolutionary tool in higher education, transcending classroom boundaries and permeating various aspects of the academic experience. While mobile applications with AI have proven to be beneficial, as evidenced by Martinez and Rodríguez [36], their potential in the university context is undeniable.

Immersive technologies, such as Virtual Reality, have solidified themselves as crucial tools in higher education, facilitating learning for students facing specific challenges, a fact supported by research like that of Rodriguez et al. [52]. In addition to these direct applications, personalized learning, especially in key areas such as languages and mathematics, has been transformed by tutoring systems and adaptive models [15, 37].

Administrative efficiency, a crucial element for the seamless functionating of university institutions, has likewise been enhanced thanks to AI, streamlining processes from admissions to academic management [47].

Finally, in the field of formative research, AI has become a catalyst, boosting the ability to analyze data on a large scale and facilitating discoveries in disciplines as diverse as medicine and engineering [53]. In summary, AI is not only redefining pedagogy in higher education but also expanding the possibilities and horizons of research and university administration.

1.4 Practical Implications and Future Trends

Artificial Intelligence serves as the guiding for the future of higher education. Projections of hybrid educational models, as outlined by Moreno et al. [39], depict a landscape where traditional classrooms merge with virtual environments, creating learning experiences tailored to the circumstances and individual needs of each student. But beyond personalized learning, AI has the potential to act as a sentinel, as Park [43]

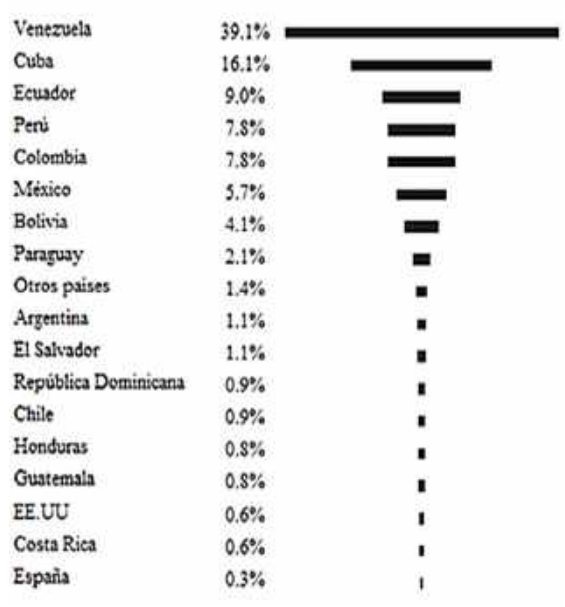


Fig. 2. Participation of teachers from Latin America and the Caribbean

suggests, identifying students in vulnerable situations early and enabling precise and timely interventions.

The educational landscape also expands into a holistic and continuous perspective. As noted by Romero and Romero [53], García [23], and Torres and Yucra [61], there is an expectation of interdisciplinary convergence, where AI acts as a bridge between disciplines, fostering continuous education that adapts to changes and demands in the professional and social environment.

On the other hand, emerging tools on the horizon, such as brain-computer interfaces mentioned by Valerdi et al. [63], could not only change but revolutionize our relationship with learning, allowing us to interact in previously unimaginable ways with educational systems.

Although the transformation promised by AI in higher education holds great promise, it is essential to maintain ongoing research and constant adaptation to these innovations.

In doing so, we ensure not only the effective implementation of these tools but also that the guiding principle remains high-quality and relevant education for the benefit of future generations.

1.5 Ethical Perspective of AI

The integration of AI in the field of education, particularly in higher education, has opened up a promising landscape in terms of personalized learning and administrative efficiency. However, its adoption has also triggered a series of ethical challenges. Concerns about data privacy, algorithmic bias, and the autonomy of the learning process have emerged strongly.

In this context, it is imperative that university institutions, when adopting AI-based solutions, balance technological innovation with a strong ethical commitment, thus ensuring an education that respects and promotes the rights and dignity of all involved.

Nuveo, an emerging Brazilian startup, highlights the need for strong digital ethics in its implementation of facial recognition technologies in the educational domain [7]. In parallel, Bujosa [9] and De Asis [16] concur on the need to establish ethical guidelines for the integration of AI in the legal domain, emphasizing the urgency of legislating ethical-legal principles.

Conversely, within the healthcare sector, AI has demonstrated revolutionary potential, from diagnosing heart diseases to managing biomedical data [20]. However, its rapid deployment in exigent scenarios, such as pandemics, highlights the imperative need for an ethical framework [12].

The interaction between humans and artificial intelligence has generated substantial discussions. In this regard, Leal [35] and Fernández [19] reaffirm the need to focus AI toward ethics and humanism, highlighting concepts such as transhumanism, bioethics, and the Charter of Fundamental Rights. García [25] and Hueso [31] emphasize the urgency of ethical governance, with the European Union striving to lead in this direction [33, 32].

Finally, literature, such as 'Más (que) humanos,' explores the potential role of AI in enhancing our ethical capacity, a topic subjected to rigorous analysis in reviews like that of Rueda [55].

With this background, a fundamental question arises: What are the ethical challenges associated with the use of ChatGPT from the perspective of university professors in Latin America? To answer this question, this study has the general objective of analyzing the ethical challenges associated with

Table 1. Average ethical challenges regarding the use of ChatGPT in higher education in Latin America

		N	Mín.	Max.	Half	Standard Deviation	Variance
Ethical Challenges	DI Privacy and data security	665	1	4	2.32	1.056	1.115
	D2 Bias and Discrimination	665	1	5	3.00	1.431	2.047
	D3 Ethics and responsibility	665	1	4	3.18	1.014	1.029
	D4 Autonomy and decision making	665	1	5	3.21	1.496	2.238
Use of ChatGPT	D1 Functions and applications	665	1	5	3.67	1.405	1.974
	D2 User experience	665	1	3	2.39	0.726	0.527
	D3 Efficiency and efficacy	665	1	3	2.21	0.803	0.645
	D4 Perceptions and attitudes	665	1	3	2.38	0.728	0.529
	D5 Impact	665	1	4	3.03	1.062	1.129

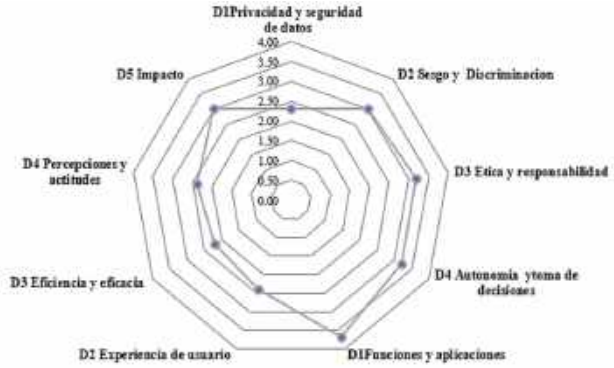


Fig. 3. Mean of ethical challenges regarding the use of ChatGPT in higher education in Latin America

the use of ChatGPT from the perspective of university professors in Latin America.

Specifically, four specific objectives were formulated, aiming to determine data privacy and security, examine bias and discrimination in use, identify ethical concerns and responsibilities related to its use, and finally, determine autonomy and ethical decision-making in the classroom.

These research endeavors aim to provide an in-depth perspective on the role of ChatGPT in education and serve as a guide for future technological adaptations and implementations in the pedagogical field.

The justification for this study stems from the urgent necessity to understand the emergence of Artificial Intelligence in higher education, which has aroused both expectations and reservations. While tools like ChatGPT offer innovative pedagogical possibilities, ethical dilemmas also arise, from data privacy to potential biases in teaching.

As Latin America is rapidly adopting these innovations, it is crucial to understand how university professors in the region perceive and address these ethical challenges. Despite its relevance, there is a gap in the literature regarding this regional and practical perspective.

This research seeks to bridge that gap by providing insights that can guide educational policies and practices. Ultimately, the goal is to ensure that the implementation of AI is conducted ethically and in line with educational values. In a world where technology and ethics converge, this research holds significant importance for education adapted to the 21st century.

2 Methodology

2.1 Type and Design

The methodology implemented in this research sought to provide a comprehensive overview of the

Table 2. Level of knowledge of ChatGPT use by teachers' age and gender

			Knowledge level about ChatGPT				
			Hight	**Medium**	**Under**	**No use**	**Total**
Age	20 - 29 años	N	3	3	10	17	33
		%	0.5%	0.5%	1.5%	2.6%	5.0%
	30 - 39 años	N	18	16	45	68	147
		%	2.7%	2.4%	6.8%	10.2%	22.1%
	40 - 49 años	N	20	28	55	105	208
		%	3.0%	4.2%	8.3%	15.8%	31.3%
	50 - 59 años	N	26	26	77	90	219
		%	3.9%	3.9%	11.6%	13.5%	32.9%
	Más de 60 años	N	4	8	19	27	58
		%	0.6%	1.2%	2.0%	4.1%	8.7%
	Total	N	71	81	206	307	665
		%	10.7%	12.2%	31.0%	46.2%	100.0%
Sex	Masculine	N	45	49	141	215	450
		%	6.8%	7.4%	21.2%	32.3%	67.7%
	Femenine	N	26	32	65	92	215
		%	3.9%	4.8%	9.8%	13.8%	32.3%
	Total	N	71	81	206	307	665
		%	10.7%	12.2%	31.0%	46.2%	100.0%

ethical challenges faced by university teachers in Latin America, particularly in the area of Artificial Intelligence (AI) integration in the educational process.

The robust nature of the methodology used lays the foundation for future research and recommendations in this domain. The research adopted a mixed approach [30], amalgamating qualitative and quantitative techniques for data collection and analysis.

This combination resulted in a richer and more complete interpretation of the phenomenon under study. With a descriptive-explanatory scope, [6] the research not only detailed the inherent characteristics of the sample, but also sought to understand and explain the relationships between the different variables. A non-experimental design was selected for the study, allowing the subjects to be observed in their natural context, without any external interventions.

2.2 Subjects

The population [11] for this study included university teachers belonging to the Latin American and Caribbean Network of Scientific Researchers (RED ICALC), with a specific inclusion of teachers from Cuba and the United States.

From this large population, a sample of 665 teachers was selected, ranging in age (from 20 to over 60 years old) and gender.

The survey reveals a predominance of teachers from Venezuela (39.1%) and Cuba (16.1%). Although Colombia and Peru have similar shares, Ecuador is slightly ahead. Despite the influence of Spain and the United States, their presence is low, with Mexico and Bolivia contributing around 5%.

Other Latin American countries range between 1% and 2.1%. A stratified probability sampling method was adopted to ensure fair and equitable representation, based on illegibility criteria such as gender, type of institution and employment status [41].

2.3 Instruments

The questionnaire was used as the primary instrument for data collection, focusing on the acquisition of quantitative data pertaining to the ethical challenges linked to AI in education. Semi-structured interview guides [11] were used to

Table 3. Ethical challenges regarding the use of ChatGPT in university education

		Use of ChatGPT							
		Efficiency of learning		Reliability of Information		Integration with Other Educational Resources		Total	
		N	%	N	%	N	%	N	%
Ethical Challenges	Concern about Bias and Discrimination	19	15.7%	21	12.7%	28	9.8%	78	11.7%
	Need for Regular Ethical Evaluations	30	24.8%	37	22.4%	74	25.8%	168	25.3%
	User in Ethical Use	40	33.1%	63	38.2%	113	39.4%	251	37.7%
	Regulation and Guidelines	32	26.4%	44	26.7%	72	25.1%	168	25.3%
	Subtotal	121	100.0%	165	100.0%	287	100.0%	665	100.0%

capture qualitative data, providing a detailed understanding of teachers' perspectives and experiences of AI in education [4].

The instruments were subjected to a validation process by experts in the field (11), ensuring high validity. Reliability was obtained through Cronbach's alpha coefficient, with values of 0.89 and 0.91 respectively. To facilitate data collection process, the digital tool Google Forms was utilized.

Data were analyzed using descriptive statistics, highlighting measures of central tendency (mean and median) and dispersion (variance and standard deviation). These metrics provided an overview of trends, such as average age of teachers, gender distribution and type of institution. In addition, inferential statistics were used to infer about the total population from the data collected from the objectives.

3 Results and Discussion

Digital tools, in particular ChatGPT, have revolutionized the way teaching and learning takes place in the classroom. However, with the power of these tools come ethical responsibilities and concerns. These concerns not only have the potential to shape educators' perceptions of the tool, but can also influence the way they employ it.

In Latin American university education, ChatGPT shows an ambivalent picture. Although there are concerns about privacy, with a mean of 2.32 on a scale of 1-4, teachers remain neutral about bias, with an average of 3 on a scale of 1-5.

The ethics of the tool are rated positively (3.18/4), but there is variability in opinions about its autonomy (3.21/5, standard deviation 1.496). Despite these reservations, teachers value the educational experience facilitated by ChatGPT, and their overall perception is favorable, predicting a promising future for the tool in academia.

In today's digital age, technological tools are reshaping the way we educate and learn. One such tool, ChatGPT, has gained prominence in the educational sphere, being hailed for its ability to interact in real time and provide answers based on a vast knowledge base.

However, with these emerging opportunities also come responsibilities and challenges. From the perspective of university teachers in Latin America, there is a need to analyze the ethical issues associated with its use.

Table 4. Data privacy regarding the use of ChatGPT in university education

		Concerns about bias and discrimination		Need for regular ethical evaluations		User perception of ethical usage		Regulation and ethical use guidelines		Total	
		N	%	N	%	N	%	N	%	N	%
D1: Data Privacy	Level of security	68	87.2%	65	38.7%	41	16.3%	0	0.0%	174	26.2%
	Level of concern for privacy	9	11.5%	85	50.6%	119	47.4%	17	10.1%	230	34.6%
	Level of user control	1	1.3%	16	9.5%	80	31.9%	40	23.8%	137	20.6%
	Security and privacy issues	0	0.0%	2	1.2%	11	4.4%	111	66.1%	124	18.6%
	Total	78	100.0%	168	100.0%	251	100.0%	168	100.0%	665	100.0%

In the study we present, we address crucial issues related to data privacy and security, explore possible biases and discriminations, identify the main ethical concerns, and examine how these tools impact autonomy and decision-making in the context of the university classroom.

Through this analysis, we aim to provide a comprehensive and critical view of the role of ChatGPT in Latin American higher education. In relation to Latin American teachers' knowledge levels regarding the use of ChatGPT in university teaching, interesting patterns emerge concerning gender and age.

Regarding gender, it becomes evident that male teachers exhibit a higher level of familiarity regarding the use of ChatGPT in university teaching compared to female teachers. 67.7% of male teachers possess some level of knowledge, while only 32.3% of female teachers have some level of knowledge.

In terms of age, a prevalent trend of higher knowledge can be observed among younger teachers. Those aged 20-29 exhibit a notably higher level of knowledge (3.1%) compared to teachers over 60 (0.6%).

Within the middle group, ages 30-59 demonstrate varying levels of knowledge, but in general, younger teachers tend to be more familiar with the use of ChatGPT in university teaching.

These findings underscore the importance of customizing training and technology promotion strategies according to teachers' gender and age to ensure the effective adoption of ChatGPT in higher education. The study aimed to understand the perception of ChatGPT in education. A 12.8% perceive biases that could affect its use, while 35.9% believe it may complicate its integration with other media. From an ethical perspective, 44.0% consider ethical evaluations essential.

It is noteworthy that 45.0% emphasize the role of the educator or student in its ethical use. Finally, although only 11.9% believe regulations have direct impact on its utilization, a significant 42.9% view regulations as essential for integrating ChatGPT into education.

The perception of biases in ChatGPT moderately influences its educational use, with 12.8% feeling that it affects classes, and 35.9% believing it limits its integration with other media. It is crucial for 44.0% to conduct periodic ethical assessments. 45.0% emphasize the user's responsibility for its ethical use.

Although only 11.9% see guidelines as determinants in its use, 42.9% consider them essential for its educational integration. Within the Latin American educational context, there is a widespread concern about biases in tools like ChatGPT.

Nearly everyone (92.3%) acknowledges the necessity of a regulatory framework to address these biases. Although the importance of ethical reviews (45.2%) and user education (20.3%) in

Table 5. Bias and discrimination regarding the use of ChatGPT in higher education

		Concerns about bias and discrimination		Need for regular ethical evaluations		User perception of ethical usage		Regulation and ethical use guidelines		Total	
		N	%	N	%	N	%	N	%	N	%
D2: Bias and discrimination	Perpetuation of bias	72	92.3%	47	28.0%	14	5.6%	0	0.0%	133	20.0%
	Capacity to reduce bias	5	6.4%	76	45.2%	51	20.3%	0	0.0%	132	19.8%
	Detect biases	0	0.0%	42	25.0%	104	41.4%	6	3.6%	152	22.9%
	Design responsibility	1	1.3%	3	1.8%	62	24.7%	33	19.6%	99	14.9%
	Responsibility of users	0	0.0%	0	0.0%	20	8.0%	129	76.8%	149	22.4%
	Total	78	100.0%	168	100.0%	251	100.0%	168	100.0%	665	100.0 %

mitigating biases is recognized, a significant proportion (41.4%) believes that detecting biases largely depends on the ethical use by the user.

However, the greatest responsibility lies with the designers and administrators of these tools, according to 76.8% of the respondents. Upon analyzing the collected data, it was determined that 68.2% of the teachers perceive ChatGPT as a catalyst for enhancing the quality of education, while 15.5% remain neutral on the matter. However, 16.3% feel that it has not had a significant impact.

Furthermore, 82% believe that, with proper training, tools like ChatGPT could revolutionize traditional pedagogy. It is worth noting that 72.5% of the participants emphasized the importance of continuous access to updates and training on this technology for its optimal use in the classroom.

Technological advancements in education have catapulted tools like ChatGPT to the center of the pedagogical debate. An overwhelming 95% of teachers emphasize the urgency of regulations addressing the impact of ChatGPT on student autonomy, while only 2.8% believe that students are already using it ethically.

Although there is confidence in ethical use by educators (10%) and students (41.4%), 38.1% highlight the need for constant ethical assessments for teachers. Despite an approval rate of 83.3% regarding the existing regulations governing decision-making with ChatGPT, only 25.5% consider that it is respected in practice, showing a mismatch between norms and reality.

4 Discussion and Conclusions

In the Latin American educational context, the application and perception of tools like ChatGPT have been the subject of scrutiny and debate. Our objectives were focused on understanding the interaction of this emerging technology with the involved stakeholders, breaking down its impacts, advantages, and challenges from pedagogical, technical, and ethical perspectives.

The perception of ChatGPT in education highlights concerns about biases and its integration with other educational media.

Despite its recognized benefits, ethical evaluation and educational mediation are essential for its optimal use. These findings align with the research by Flores and García [22] on ethics in educational AI and with Mosquera et al. [40], which examines the fusion of AI and ICT in music education.

However, Brochado [8] presents a different angle, focusing on the ethical dilemmas when AI reaches human-level efficiency. Parga [42] emphasizes that AI, including tools like ChatGPT, should

Table 6. Ethics and responsibility regarding the use of ChatGPT in higher education

		Concerns about bias and discrimination		Need for regular ethical evaluations		User perception of ethical usage		Regulation and ethical use guidelines		Total	
		N	%	N	%	N	%	N	%	N	%
D3: ethics and responsibility	Ethical challenges for teaching	59	75.6%	14	8.3%	0	0.0%	0	0.0%	73	11.0%
	Ethical user use	14	17.9%	46	27.4%	13	5.2%	0	0.0%	73	11.0%
	Ethical use guidelines	5	6.4%	85	50.6%	86	34.3%	3	1.8%	179	26.9%
	Ethical use regulation	0	0.0%	23	13.7%	152	60.6%	165	98.2%	340	51.1%
	Total	78	100.0%	168	100.0%	251	100.0%	168	100.0%	665	100.0%

Table 7. Autonomy and decision-making regarding the use of ChatGPT in higher education

		Concerns about bias and discrimination		Need for regular ethical evaluations		User perception of ethical usage		Regulation and ethical use guidelines		Total	
		N	%	N	%	N	%	N	%	N	%
D4: Autonomy and decision making	Impact on student autonomy	74	94.9%	49	29.2%	7	2.8%	0	0.0%	130	19.5%
	Freedom of the educator	4	5.1%	64	38.1%	25	10.0%	0	0.0%	93	14.0%
	Freedom of the student	0	0.0%	45	26.8%	104	41.4%	6	3.6%	155	23.3%
	Freedom of use	0	0.0%	9	5.4%	51	20.3%	22	13.1%	82	12.3%
	ChatGPT compatibility and decision making	0	0.0%	1	0.6%	64	25.5%	140	83.3%	205	30.8%
	Total	78	100.0 %	168	100.0 %	251	100.0 %	168	100.0 %	665	100.0 %

prioritize humanism and fundamental rights. In summary, ChatGPT has transformative potential in education, but its adoption in Latin America requires proactively addressing ethical and technical concerns, emphasizing proper training for its implementation. Incorporating ChatGPT into education has brought forth ethical and bias concerns.

Despite its limited adoption, there is a pronounced need for ongoing ethical evaluations

and active user engagement to ensure its proper application. This perspective aligns with the study by Rebolledo and Abufarde [48], where they optimized processes through simulation, emphasizing the importance of continuous assessment.

Tocto et al. [60] also employed AI, in this case, to predict student graduation, showcasing the analytical potential of such tools. In contrast, Piteira et al. [44] present a divergent perspective, stating that AI itself poses ethical dilemmas, focusing on the moral issues of "thinking" machines and citing research by ACM and IEEE.

Concerns regarding biases in ChatGPT within the Latin American educational context emphasizes the necessity for both regulations and ethical education. This viewpoint resonates with Torres and Yucra [61], who identified negative perceptions among students towards virtual classes.

Vlasova et al. [68] underscores the importance of preparing educators in AI usage, proposing an adaptive training system, highlighting the necessity of user training. Conversely, Solé [57] demonstrates a discrepancy between ethical self-regulation and legal regulation of AI, with the US and the EU adopting divergent approaches. The variety of these approaches underscores the urgency of a transdisciplinary perspective to address ethics and regulation in AI.

The growing demand for regulations surrounding technologies like ChatGPT in education reflects an increasing ethical awareness. While the community values the potential of these tools, it recognizes that the human dimension is vital.

Rodríguez [50, 51] advocates for the development of sustainable AI, considering its environmental and cultural impact, drawing inspiration from ethics of responsibility and care. In contrast, Cantarini [10] advocates for a "poietic" relationship with AI, departing from linear paradigms and promoting symbiotic coexistence and technodiversity.

Despite the differences, both research studies emphasize the interconnection between ethics and technology. Ethics and technology in education are inextricably linked. Guidelines should empower and educate users, recognizing the centrality of ethics and humanity in the era of AI. ChatGPT, with its promising educational potential, highlights the necessity for regulations to safeguard student autonomy in light of ethical concerns. Vásquez et al. [65] show parallel concerns in the Chilean legal context, highlighting challenges in implementing educational programs that combine artificial intelligence and law.

While both of Vásquez's studies coincide in recognizing challenges related to technology adoption in conventional sectors, García [25] presents another dimension by discussing AI self-programming and its unpredictable outcomes. This factor further underscores the urgency of ethical controls.

In summary, while technology like ChatGPT can transform education, it is imperative to have regulations that ensure its ethical use, striking a balance between innovation and human values. Neglecting to effectively address these identified concerns could lead to an educational landscape characterized by biases, compromised privacy, and restricted student autonomy.

While our sample is representative, it is not all-encompassing. Perceptions and experiences may diverge depending on specific contexts, and our research may not capture all perspectives within the diversity of Latin American educational environments. It is imperative that future research delve into qualitative methods and explore the long-term impact of tools like ChatGPT, assessing how they influence educational outcomes and student well-being.

5 Conclusion

The incorporation of advanced tools like ChatGPT in Latin American university education holds significant potential. However, the findings reveal shared concerns among educators regarding inherent biases and ethical implications of the technology. While a strong demand for clear regulations and consistent ethical assessments is perceived, there is also a recognition of collective responsibility, both from those designing these tools and from end-users. Ultimately, to fully harness the benefits of ChatGPT, it is essential to balance technological innovation with ethical imperatives, ensuring inclusive and empowering education for all.

References

1. **Ali, M., Shamsan, A., Hezam, A., Mohammed, A. (2023).** Impact of ChatGPt on learning motivation. Journal of English Studies in Arabia Felix, Vol. 2, No. 1, pp. 41–49. DOI: 10.56540/jesaf.v2i1.51.
2. **Arce-Guerrero, S., González-Jimenez, D. A. (2022).** Superando estereotipos de género en la enseñanza las ingenierías. El caso de un curso de introducción a las ingenierías, Encuentro Internacional de Educación en Ingeniería, EIEI ACOFI, DOI: 10.26507/paper.2156.
3. **Arevalillo, M., Moreno, S., Cavero, V. (2011).** A computer architecture to use AIBO robots to teach artificial intelligence. Revista Iberoamericana de Tecnologías del aprendizaje, Vol. 6, No. 1, pp. 1–9.
4. **Arias-Gonzalez, J. L. (2020)**. Técnicas e instrumentos de investigación científica. Primera edición, Editorial Enfoques Consulting Eirl. pp. 1–171.
5. **García-Benítez, V. H., Ruvalcaba-Gómez, E. A. (2021).** Analysis of national artificial intelligence strategies in Latin America: A study of the ethics and human rights approaches. Revista de Gestion Publica, Vol. 10, No. 1, pp. 5–32. DOI: 10.22370/rgp.2021.10.1.3151.
6. **Bernal-Torres, C. A. (2010).** Metodología de la investigación administración, economía, humanidades y ciencias sociales. Pearson Educación, pp. 1–305.
7. **Marques-Castelo-Branco-Biondi, G., Kemmer-Cernev, A. (2023).** Nuveo: digital ethics and artificial intelligence for real world challenges. Revista de Administracao Contemporanea, Vol. 27, No. 3. DOI: 10.1590/1982-7849rac2023220063.
8. **Brochado, M. (2023).** Artificial intelligence and ethics: A dialogue with Lima Vaz. Kriterion: Revista de Filosofia, Vol. 64, No. 154, pp. 75–98. DOI: 10.1590/0100-512x2023n15404mb.
9. **Bujosa-Vadell, L. (2022).** Ethics and artificial intelligence: a look from the jurisdictional process. Revista Eletronica de Direito Processual, Vol. 23, No. 1, pp. 733–768. DOI: 10.12957/redp.2022.64391.
10. **Cantarini, P. (2022).** Por Uma Ética Da Inteligência Artificial Com Base Na Poiética Revista Juridica, Vol. 4, No. 71, pp. 892–912. DOI: 10.26668/revistajur.2316-753x.v4i71.6170
11. **Carrasco-Diaz, S. (2016).** Metodología de investigación científica. Pautas metodológicas para diseñar y elaborar el proyecto de investigación (Cuarta). Editorial San Marcos. pp. 1–476.
12. **Castrillón, O. D., Rodríguez, M. D. P., Leyton, J. D. (2008).** Ethics and artificial intelligence, need or urgency? CISCI´08 Septima Conferencia Iberoamericana en Sistema, Cibernetica e Informatica 5to Sieci 2008, 3er Simposium Internacional en Comunicacion del Conocimiento y Conferencias, CCC 2008, Memorias, Vol. 3, pp. 121–126.
13. **Cerda, M., Matero, D., Álvarez, J. (2022).** Heritage education and virtual archaeology applications in museums and archaeological sites. Human Review Revisat Internacional de Humanidades, Vol. 15, No. 4, pp. 1–14. DOI: 10.37467/revhuman.v11.4254.
14. **Chicaiza, R. M., Camacho, I., Ghose, G., Castro, I., Gallo, V. (2023).** Aplicaciones de ChatGPT como inteligencia artificial para el aprendizaje de idioma inglés: Avances, desafíos y perspectivas futuras applications of ChatGPT as artificial intelligence for english language learning: advances, challenges, and future perspectives. Latam Revista Latinoamericana de Ciencias Sociales y Humanidades, Vol. 4, No. 2, pp. 2610–2628. DOI: 10.56712/latam.v4i2.781.
15. **Cotán, A., Ruiz, A. M., Álvarez, K. (2022).** Innovando en educación: la foto-elicitación como estrategia de aprendizaje en las instituciones de educación superior. Márgenes Revista de Educación de La Universidad de Málaga, Vol. 3, No. 2, pp. 137–153. DOI: 10.24310/mgnmar.v3i2.14104.
16. **De-Asis, M. (2023).** Ethics of artificial intelligence applied to judicial process. Cuadernos Electrónicos De Filosofía del Derecho, Vol. 48, pp. 60–79. DOI: 10.7203/cefd.48.25389.

17. **Donato, H., Escada, P., Villanueva, T. (2023).** The transparency of science with ChatGPT and the emerging artificial intelligence language models: Where should medical journals stand? Acta Médica Portuguesa, Vol. 36, No. 3, pp. 147–148. DOI: 10.20344/amp.19694.
18. **Imbaquingo-Esparza, D. E., Hernández-Obando, C. C. (2019).** Estudio de la seguridad en big data, privacidad y protección de datos mediante la ISO/IEC 27007:2017- Aplicado a los datos académicos de la Universidad Técnica del Norte, Universidad Técnica del Norte. En Repositorio Digital Universidad Técnica Del Norte.
19. **Fernández-Fernández, J. L. (2021).** Towards digital humanism from a common denominator for cyber ethics and artificial intelligence (AI) Ethics. Disputatio (Spain), Vol. 10, No. 17, pp. 107-130. DOI: 10.5281/zenodo.5136247.
20. **Marques-de-Souza-Filho, E., de-Amorim-Fernandesn, F., de-Assis-Pereira, N. C., Tinoco-Mesquita, C., Altenburg-Gismond R., (2020).** Ethics, artificial intelligence and cardiology. Arquivos Brasileiros de Cardiologia, Vol. 115, No. 3, pp. 579–583. DOI: 10.36660/abc.20200143.
21. **Flores-Masias, E. J., Livia-Segovia, J. H., Casique, A. G., Dávila-Díaz, M. E. (2022).** Sentiment analysis with artificial intelligence to improve the teaching-learning process in the virtual classroom. Publicaciones de la Facultad de Educación y Humanidades del Campus de Melilla, Vol. 53, No. 2, pp. 201–216. DOI: 10.30827/publicaciones. v53i2. 26825
22. **Flores-Vivar, J. M, García-Peñalvo, F. J. (2023).** Reflections on the ethics, potential, and challenges of artificial intelligence in the framework of quality education (SDG4). Reflexiones sobre la ética, potencialidades y retos de la Inteligencia Artificial en el marco de la Educación de Calidad (ODS4). Revista Científica de Comunicación y Educación Comunicar, Vol. 30, No. 74, pp. 35–44. DOI: 10.3916/c74-2023-03.
23. **García-Peñalvo, F. J. (2023).** La percepción de la inteligencia artificial en contextos educativos tras el lanzamiento de ChatGPT: disrupción o pánico. Education in the Knowledge Society (EKS), Vol. 24, pp. e31279–e31279. DOI: 10.14201/eks.31279.
24. **García-Torres, I. A., Castillo-León, R. E., Taranto-Vera, G. J. (2023).** Efectividad del aprendizaje virtual en comparación con el aprendizaje presencial usando una herramienta de IA basada en Machine Learning. RECIAMUC, Vol. 7, No. 2, pp. 210–218. DOI: 10.26820/reciamuc/7.(2).abril.2023. 210-218.
25. **García-Vigil, J. L. (2021).** Reflections around ethics, human intelligence and artificial intelligence. Gaceta Médica de México, Vol. 157, No. 3, pp. 298–301. DOI: 10.24875/gmm. m21000561.
26. **Gómez-Diago, G. (2022).** Perspectives to address artificial intelligence in journalism teaching. A review of research and teaching experiences. Revista Latina de Comunicación Social, Vol. 80, pp. 29–45. DOI: 10.4185/rlcs-2022-1542.
27. **González-García, M. (2022).** La educación del siglo XXI. Claves para la personalización del aprendizaje a través de la neurodidáctica. Padres y maestros. Journal of Parents And Teachers, Vol. 389, pp. 36–42. DOI: 10.14422/ pym.i389.y2022.006.
28. **Grenouilloux, A. (2023).** Digitalization, automation in psychiatry: what human guarantee? Information Psychiatrique, Vol. 99, No. 2, pp. 83–90. DOI: 10.1684/ipe.20 23.2549.
29. **Gruson, D. (2021).** Big data, artificial intelligence and laboratory medicine: Time for integration. Advances in Laboratory Medicine, Vol. 2, No. 1, pp. 5–7. DOI: 10.1515/almed-2021-0014.
30. **Hernández, R., Fernández, C., Baptista, P. (2014).** Metodología de la investigación México: MCGRAW-HILl. Vol. 6, pp. 102–256.
31. **Cotino-Hueso, L. (2019).** Ethics in design for the development of an artificial intelligence, trustworthy robotics and big data and their utility for the law. Revista Catalana Dret Públic, Vol. 58, No. 29. Pp. 29–48. DOI: 10.2436/rcdp. i58.2019.3303.

32. **Jarrín, M. T. (2021).** Eu and ethics governance of the artificial intelligence: Artificial intelligence and diplomacy. Cuadernos Salmantinos De Filosofía, Vol. 48, pp. 213– 234.
33. **Martín-Jiménez, F. J. (2023).** Inteligencia artificial y ética: hacia una aplicación de los principios éticos en el ámbito de la UE. Cuadernos Europeos de Deusto, Vol. 68, pp. 89–115. DOI: 10.18543/ced.2699.
34. **Juarez, G. E., Yelamos-Caceres, M., Menendez, F. D., Lafuente, C., Franco, I., Perez, J. O., Rivero, C. R. (2019).** Integration of relational databases in ethical decision-making for autonomous vehicles. IEEE Biennial Congress of Argentina, Argencon 2018. pp. 1–6. DOI: 10.1109/argencon.2018.8646063.
35. **Zavala-Leal, T. D. (2021).** La ética en inteligencia artificial desde la perspectiva del derecho. Via Inveniendi Et Iudicandi, Vol. 16, No. 2. DOI: 10.15332/19090528.6785.
36. **Rodriguez-Umaña, L., Martinez-Baquero, J. (2022).** Uso de aplicaciones móviles como herramienta de apoyo tecnológico para la enseñanza con metodología STEAM. Revista Politécnica, Vol. 18, No. 36, pp. 75–90. DOI: 10.33571/rpolitec.v18n36a6.
37. **Medrano, J. F., Tererina, M., Castillo, C. (2019).** Empleo de chatbots educativos como recurso complementario en las prácticas docentes. Researchgate. XXI Workshop de Investigadores en Ciencias de la Computación (WICC 2019, Universidad Nacional de San Juan).
38. **Al-Worafi, Y. M., Hermansyah, A., Wen-GohK, Chiau-Ming, I. (2023).** Artificial Intelligence Use in University: should We Ban ChatGPT? DOI: 10.20944/preprints202302.0400.v1.
39. **Moreno-Zagal, M., Cruz-Arizmendi, E. T., Muñoz-Muñoz, I. (2023).** La nueva tutoría en educación superior dentro de los modelos híbridos. Revista de Educación y Desarrollo, Vol. 64, pp. 17–26.
40. **Paitán, H. Ñ., Mejía, E. M., Ramírez, E. N., Paucar, A. V. (2014).** Metodología de la investigación cuantitativa-cualitativa y redacción de la tesis. Ediciones de la U.
41. **Otero-Parga, M. (2023).** ¿Puede la inteligencia artificial sustituir a la mente humana? Implicaciones de la IA en los derechos fundamentales y en la ética. Anales de la Catedra Francisco Suarez, Vol. 57, pp. 39–61. DOI: 10.30827/acfs.v57i.24710.
42. **Ho-Park, S. (2023).** Authorship policy of the Korean journal of radiology regarding artificial intelligence large language models such as ChatGTP. Korean Journal Radiology, Vol. 1, No. 2023, DOI: 10.3348/kjr.2023.0112.
43. **Piteira, M., Aparicio, M., Costa, C. J. (2019).** Ethics of artificial intelligence: Challenges. 2019 14th Iberian Conference on Information Systems and Technologies (CISTI). DOI: 10.23919/cisti.2019.8760826.
44. **Medina-Quero, J., Ruiz-Lozano, M. D., Serrano-Chica, J. M., Espinilla-Esteves, M. (2020).** Integración de conceptos tecnológicos emergentes en educación superior a través de aprendizaje basado en proyectos. En Tecnologías Emergentes y Estilos de Aprendizaje para la Enseñanza, pp. 112–122.
45. **Quezada-Castro, G. A., Castro-Arellano, M. P., Quezada-Castro, M. P. (2022).** Artificial intelligence and legal education: Its incorporation during the Covid-19 pandemic. Revista Venezolana de Gerencia, Vol. 27, No. 8, pp. 750–764. DOI: 10.52080/rvgluz.27.8.2.
46. **Ramirez-Chavez, V. G. (2021).** Pensamiento crítico y su influencia en la autonomía del aprendizaje en estudiantes de secundaria. IGOBERNANZA, Vol. 4, No. 14, pp. 197–203. DOI: 10.47865/igob.vol4.2021.121.
47. **Rebolledo, M. M., Abufarde, F. B. (2001).** The use of artificial intelligence for the optimization of a simulation model applied to a radiata pine manufacturing process. Maderas: Ciencia y Tecnología, Vol. 3, No. 1-2, pp. 52– 62.
48. **Rincón, J., Vila, M. (2021).** Modelo predictivo multivariable en tiempo real para predecir el desempeño de los estudiantes, en programas virtuales de posgrado, empleando inteligencia artificial. American Journal of Distance

Education, Vol. 35, No. 4, pp. 307–328. DOI: 10.1080/08923647.2021.1954839.

49. **Terrones-Rodríguez, A. I. T. (2022).** Ëtica para la inteligencia artificial sostenible. ARBOR, Vol. 198, No. 806. DOI: 10.3989/arbor.2022.806013.
50. **Terrones-Rodríguez, A. L. (2022**). Inteligencia artificial sostenible y evaluación ética constructiva. ISEGORIA, Vol. 67. DOI: 10.3989/isegoria.2022.67.10.
51. **Rodríguez-Cano, S., Delgado-Benito, V., Casado-Muñoz, R., Cubo-Delgado, E., Ausín-Villaverde, V., Santa-Olalla-Mariscal, G. (2021).** Tecnologías emergentes en educación inclusiva: realidad virtual y realidad aumentada. Proyecto europeo FORDYSVAR. INFAD: Revista de Psicología, International Journal of Developmental and Educational Psychology, Vol. 2, pp. 443–450. DOI: 10.17060/ijodaep.2021.n1.v2.2093.
52. **Romero, S., Romero, E. (2020).** Inteligencia artificial y medicina. Revista de Cirugia de Galicia, Vol. 6, No. 7, pp. e1–e2.
53. **Rosano-Reyes, E., Corona-Jiménez, M. A. (2022).** Proximidad de las decisiones gerenciales a la responsabilidad social universitaria humanista. Revista del Centro de Investigación de la Universidad La Salle, Vol. 15, No. 58, pp. 33–60. DOI: 10.26457/recein.v15i58.2900.
54. **Rueda, J. (2023).** ¿Automatizando la mejora moral humana? La inteligencia artifical para la ética. Nota crítica sobre Lara, F. y J. Savulescu (Eds.) (2021), más (que) humanos. Biotecnología, inteligencia artificial y ética de la mejora, Madrid: Tecnos, Daimon Revista Internacional de Filosofía, No. 89, pp. 199–209. DOI: 10.6018/daimon.508771.
55. **Shafaq-Shah, S., Ali-Shah, A., Memon, F., Ahmad-Kemal, A., Soomro, A. (2021).** Aprendizaje en línea durante la pandemia de covid-19: aplicación de la teoría de la autodeterminación en la 'nueva normalidad'. Revista de Psicodidáctica, Vol. 26, No. 2, pp. 169-178. DOI: 10.1016/j.psicod.2020.12.004.
56. **Solé, J. P. (2022).** The relationships between artificial intelligence, regulation and ethics, with special atention to the public sector. Revista General de Derecho Administrativo, Vol. 61, pp. 1–29.
57. **Aguayo, C., Taheri, A., Steagall, M. (2021).** Diseño inmersivo encarnado para el aprendizaje basado en la experiencia y la autoiluminación. Link Symposium Abstracts Vol. 2, No. 1.
58. **Talan, T., Kalinkara, Y. (2023).** El papel de la inteligencia artificial en la educación superior: evaluación de ChatGPT para el curso de anatomía. Revista internacional de sistemas de información de gestión y ciencias de la computación, Vol. 7, No. 1, pp. 33–40. DOI: 10.33461/uybisbbd.1244777.
59. **Tocto, P., Huamaní, G. T., Villacorta, E. (2022).** Application of artificial intelligence in the management of a public university in Peru: A case of supervised machine learning using neural networks to classify if an engineering student would graduate in 5 years. Proceedings of the LACCEI international Multi-conference for Engineering, Education and Technology Latin American and Caribbean Consortium of Engineering Institutions. Vol. 2022. DOI: 10.18687/LACCEI2022.1.1.565.
60. **Torres, F., Yucra, Y. J. (2022).** Artificial intelligence techniques in assessment of virtual education by University students. Human Review, International Humanities Review, Revista Internacional de Humanidades, Vol. 11. DOI: 10.37467/revhuman.v11.3853.
61. **UNESCO (s. f.).** Inteligencia artificial en la Educación. Unesco. https://es.unesco.org/themes/tic-educacion/inteligencia-artificial
62. **Valerdi, L., Villarruel, M., García, J. (2020).** Interfaces cerebro-computadora: conceptualización, retos de rediseño e impacto social. Revista Mexicana de Ingeniería Biomédica, Vol. 40, No. 3. DOI: 10.17488/rmib.40.3.8.
63. **Varona, D. (2018**). The ethical responsibility of the systems designer in artificial intelligence. Revista de Occidente, No. 446, pp. 104–114.
64. **Contreras-Vásquez, P., Azuaje-Pirela, M., Díaz-Fuenzalida, J. P., Bedecarratz-Scholz, F., Bozzo-Hauri, S., Finol-González, D. (2022).** Teaching and learning law and artificial

intelligence in Chile: On the minor in Artificial Intelligence and Law at the Universidad Autónoma de Chile. Vol. 8, No. 2, pp. 281–302. DOI: 10.5354/0719-5885.2021.64456.

65. **Villamor, P. (2023).** La política de la educación en la enseñanza universitaria: una reflexión sobre su relevancia y una propuesta sobre su contenido. Teoría de la Educación. Revista Interuniversitaria, Vol. 35, No. 2, pp. 121–138. DOI: 10.14201/teri.31071.
66. **Vivanco, J.,Tocto, J., Mogrovejo, J., León, F., Vivanco, C. (2023).** Herramientas web 2.0 en la enseñanza aprendizaje de matemáticas. Una revisión bibliográfica. LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades, Vol. 4, No.2, pp.878-901. DOI: 10.56712/latam.v4i2.657.
67. **Vlasova, E. Z., Avksentieva, E. Y., Goncharova, S. V, Aksyutin, P. A. (2019).** Artificial intelligence - The space for the new possibilities to train teachers. Espacios, Vol. 40, No. 9.
68. **Yturralde-Villagomez, J., Trejo-Alarcon, J. E., Chiquito-Peñaranda, D. A., Rodríguez-Lopez, W. A. (2020).** El empleo de las aplicaciones de las tecnologías de la información y las comunicaciones en el proceso enseñanza-aprendizaje en la educación superior. Revista Dilemas Contemporaneos, VII. DOI: 10.46377/dilemas.v34i1.2218.
69. **Zhai, X. (2022).** ChatGPT user experience: implications for education. SSRN. DOI: 10.1109/MCE.2022.

Article received on 31/01/2024; accepted on 18/04/2024.
**Corresponding author is Marina Fernández-Miranda.*

ISSN 2007-9737

Mental Illness Classification on Social Media Texts Using Deep Learning and Transfer Learning

Muhammad Arif[1], Iqra Ameer[2], Necva Bölücü[3], Grigori Sidorov[1,*], Alexander Gelbukh[1], Vinnayak Elangovan[3]

[1] Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico

[2] The Pennsylvania State University at Abington, Division of Science and Engineering, USA

[3] Commonwealth Scientific and Industrial Research Organisation, Data61, Australia

{arifmuhmmand, necvaa}@gmail.com, {iqa5148, vue9}@psu.edu, {sidorov, gelbukh}@cic.ipn.mx

Abstract. Given the current social distance restrictions across the world, most individuals now use social media as their major medium of communication. Due to this, millions of people suffering from mental diseases have been isolated, and they are unable to get help in person. They have become more reliant on online venues to express themselves and seek advice on dealing with their mental disorders. According to the World Health Organization (WHO), approximately 450 million people are affected. Mental illnesses, such as depression, anxiety, etc., are immensely common and have affected an individual's physical health. Recently, Artificial Intelligence (AI) methods have been presented to help mental health providers, including psychiatrists and psychologists, in decision-making based on patients' authentic information (e.g., medical records, behavioral data, social media utilization, etc.). AI innovations have demonstrated predominant execution in numerous real-world applications, broadening from computer vision to healthcare. This study analyzed unstructured user data on the Reddit platform and classified five common mental illnesses: depression, anxiety, bipolar disorder, ADHD, and PTSD. In this paper, we proposed a Transformer model with late fusion methods to combine the two texts (title and post) of the dataset into the model to detect the mental disorders of individuals. We compared the proposed models with traditional machine learning, deep learning, and transfer learning multi-class models. Our proposed Transformer model with the late fusion method outperformed (F1 score = 89.65) the state-of-the-art performance (F1 score = 89 [35]). This effort will benefit the public health system by automating the detection process and informing the appropriate authorities about people who need emergency assistance.

Keywords. Mental illnesses classification, transformer, late fusion, machine learning, deep learning, transfer learning, Reddit.

1 Introduction

According to certain studies, mental illness can impair a person's physical health as well as her/his intellect, feelings, and behavior (or all three) [50, 32].

450 million people are affected by mental health problems such as depression, schizophrenia, attention-deficit hyperactivity disorder (ADHD), autism spectrum disorder (ASD), etc. [50]. Early diagnosis of mental illness is a fundamental step in better understanding mental health problems and providing care.

Mental illness is usually diagnosed based on self-reporting by individuals in specific surveys

Table 1. Sample instances of Reddit corpus

No.	Reddit Post	Label
1	all the ideas that normally disappear as soon as we reach for a writing device will be captured and started. imagine all the projects we will begin and never finish!	ADHD
2	i know this is long and i don't know if a lot of people will read this but i really just want to help. i had 2 panic attacks over the end of february and first day of march. i went to the doctor and had my blood work	Anxiety
3	for example, did you ever notice that you had manic, hypomanic, depressive, etc. episodes? did you ever notice that sometimes you were ßad" and other times you were ëxcessively happy"? i'm in a sticky	Bipolar
4	i just feel so trapped and i *have* to do something about it. i don't know where i'll go or what i'll do to get by. i just can't stay here any longer.	Depression
5	this is probably going to incite a lot of disagreement, maybe even anger, but that's okay; i'm going to say it anyway. anyone else tired of being told that just talking about your problems will solve your ptsd?	PTSD
6	synesthesia. what is synesthesia? according to google, ßynesthesia is a condition in which one sense (for example, hearing) is simultaneously perceived as if by one or more additional senses such as sight.	None

designed to diagnose specific patterns of feelings or social interactions, in contrast to the diagnosis of other chronic illnesses, which are based on tests and measurements in research settings [19].

In these uncertain times, with COVID-19 torments the world, many people have indicated clinical anxiety or depression. This could be due to lockdown, limited social activities, higher unemployment rates, economic depression, and work-related fatigue.

American Foundation for Suicide Anticipation reported that individuals encounter anxiety (53%) and sadness (51%) more regularly now than before COVID-19 was widespread. Within the past decade, social media has changed social interaction.

In addition to sharing data and news, people share their daily activities, experiences, hopes, emotions, etc., generating reams of data online.

This textual data provides information that can be utilized to design systems to predict people's mental health. Moreover, the current limited social interaction state has forced people to express their thoughts on social media.

In addition, because social interaction is currently limited, people are compelled to express their thoughts on social media. It gives people an open stage to share their opinions with others to find help [35].

Studies that address mental illness primarily utilized deep learning [42, 35] and traditional machine learning [36, 8] models. Recently, Transformer models [47] have gained attention with improvements in Natural Language Processing (NLP) [11, 16, 49] and Computer Vision (CV) [51, 24].

In this work, we adopted the Transformer model that is encoder part of the vanilla Transformer [47] to encode multi text (title and post) simultaneously. We theorize that encoding multiple texts with the same model can improve the quality of the mental illness problem.

We also conducted extensive experiments on late fusion methods to merge the outputs of the proposed model efficiently. We also applied traditional Machine Learning (ML), Deep Learning (DL), and Transfer Learning (TL) approaches to compare the proposed model for automatically detecting mental disorders in social media texts.

Used reddit.com[1] user data proposed by Murarka and Radhakrishnan [35] to determine

[1] https://www.reddit.com/ Last visited: 25-09-2023

Table 2. Number of posts for each mental illness classes in the train/dev/test datasets

Class	Train	Dev	Test
ADHA	2,465	248	248
Anxiety	2,422	248	248
Bipolar	2,407	248	248
Depression	2,450	248	248
PTSD	2,001	248	248
None	1,982	248	248
Total	13,727	1,488	1,488

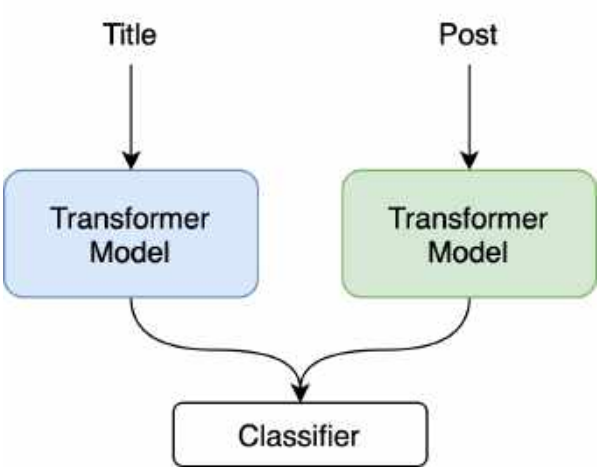


Fig. 1. Late fusion for the mental illness problem

mental illness, Table 1 represents instances of the dataset. The rest of the paper is structured as follows: Section 2 describes the studies on mental illness in literature. Sectio 3 explains the problem and gives dataset insights.

Section 4 gives details of the methodology applied to detect mental disorders with baseline models. Section 5 presents results and their analysis. Section 6 concludes the paper with possible future work.

2 Related Work

Recently, individuals have been using social media to communicate and seek advice on mental health issues. This has motivated researchers to take the information and apply various NLP and ML approaches to help individuals who may want assistance. Initially, many researchers have focused on Twitter text [37, 7, 10], later on the focus has shifted on Reddit platform [25, 17, 7, 52].

A wide range of approaches has been applied to mental health text analysis, from traditional ML to advanced DP. ML points to creating computational algorithms or statistical models capable of extracting hidden patterns from data [39, 44].

For a long time, an increasing number of ML models have been created to analyze healthcare data [36, 8]. Traditional ML approaches require a significant amount of feature engineering for ideal performance, an essential step for most application scenarios to obtain excellent performance and time [15].

Contextual content is created using words. Important insights into text classification can be gained from its structure and order [6, 2]. In the literature, several researchers have extracted the word n-grams to classify user content in social media. [25] used the word n-grams to detect mental illness from Reddit posts.

Another study [23] utilized word n-grams to generate and evaluate artificial mental health records for NLP. According to Coppersmith et al. [10], they employed character-level language models to see how probable a user with mental health concerns would create a series of characters.

Benton et al. [7] determined different types of mental health disorders by applying neural MTL, regression, and multi-layer perceptron single-task learning (STL) models.

Abussa et al. [1] trained the Support Vector Machines to distinguish 200 text messages into two classes: “ADHD or not.” The most crucial step was eliminating the acronym ADHD from the messages before learning, and further information concerning attention disorders was removed from the texts.

The goal was to see how well the Support Vector Machine learns when keywords and even semantically relevant material are unavailable. Deep feed-forward neural network has outperformed typical ML models in a variety of data mining tasks [5, 3, 2, 4], and it has been used

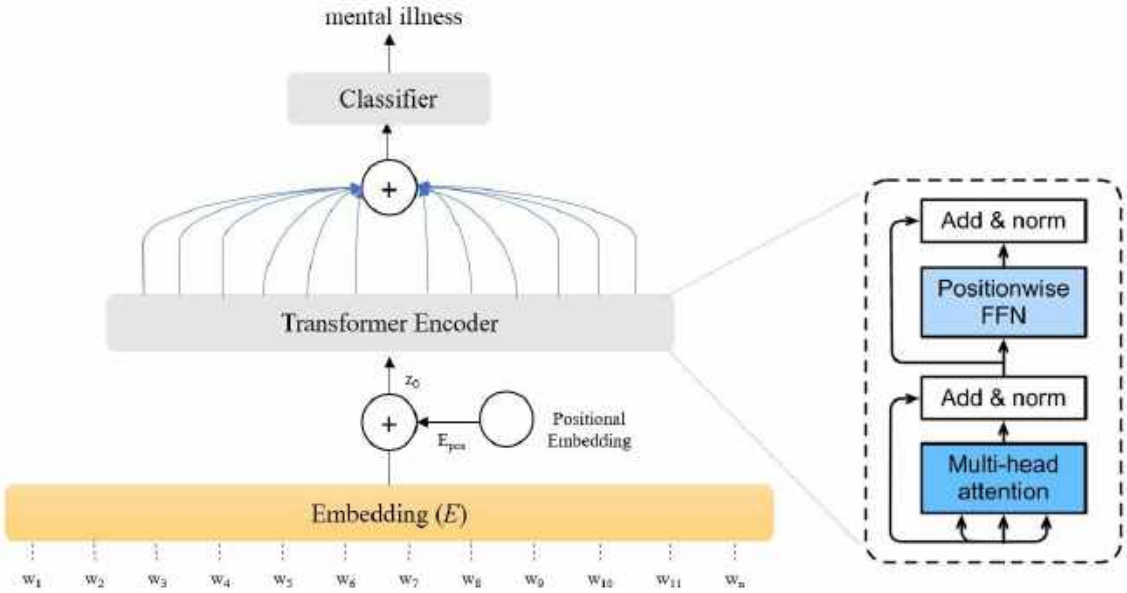


Fig. 2. Overview of the transformer architecture for mental illness problem

in the study of clinical and genetic data to predict mental health disorders.

To diagnose depression, Orabi et al. [37] used word embeddings in combination with a range of neural network models such as CNNs and RNNs. To conduct binary classification on mental health textual posts, Gkotsis et al. [17] used Feed Forward Neural Networks, CNNs, traditional ML such as Support Vector Machine, and Linear classifiers. Sekulic and Strube [41] detected depression, ADHD, anxiety, and other types of mental illnesses by training a binary classifier for each disease with Hierarchical Attention Networks.

The most recent work on this was a CNN-based classification model Kim et al. [25]. The team trained a separate binary classifier for each type of mental disorder to conduct the detection. Hu and Sokolova [21] found the potential factors to influence a person's mental health during the Covid-19 pandemic by applying ML classifiers such as Naive Bayes (NB), Logistic Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), and Gradient Boosting (GB).

They have also presented an analysis of the feature selection technique called LIME (Local, Interpretable Model-agnostic Explanations) [40]. In a recent study, Shatte et al. [42] applied ML techniques to the mental health domain.

They reviewed the literature using four key application domains—detection and diagnosis, prognosis, treatment and support, public health applications, and research and clinical administration. Another research examined the recently developed field of DL methods in psychiatry. They concentrated on DL and integrated statistical ML correlations with semantically interpretable computer models of brain dynamics or behaviour [14].

A variety of cutting-edge NN models were employed in DL-based methods. Shared task CLPsych[2] series played a significant part in developing mental health detection. CNN, RNN, LSTM, and BiLSTM were found to be the most commonly applied models.

In today's research world, TL is extremely important. Researchers attempt to acquire greater accuracy and performance in several research studies by using several types of transformers. Murarka et al. [35] examined three approaches for identifying and diagnosing mental illness

[2]clpsych.org/ Last visited: 25-09-2023

Table 3. Parameter settings of models

HyperParameter	CNN	LSTM	BiLSTM	Transformer
learning rate	$1e-5$	$1e-5$	$1e-5$	$1e-5$
batch size	16	16	16	16
# of LSTM layers	-	1	1	-
hidden units	-	250	250	250
dropout	0.1	0.2	0.2	0.2
# of Kernel	2 (3,4)	-	-	-
# of layers	-	-	-	2
# of hidden	-	-	-	2
d model	-	-	-	2
dropout	-	-	-	0.2
heads	-	-	-	2

Table 4. Results with late fusion methods

Method	Precision	Recall	F1 score
Concatenation	**89.86**	**89.58**	**89.65**
Average	88.06	87.70	87.78
Weighted Average	86.68	85.69	85.85
Maximum	87.81	87.50	87.60
Minimum	88.04	87.84	87.86

on the Reddit dataset, including LSTM, BERT, and RoBERTa.

RoBERTa outperformed the other two methods. Dhanalaxmi et al. [12] employed RoBERTa to categorize COVID-19-related informative tweets, and their method yielded the best results. Mathur et al. [33] applied LSTM with an attention mechanism to estimate suicidal intent using temporal psycholinguistics.

Shickel et al. [43] utilized a deep transfer learning model to predict emotional valence in mental health text and achieved the highest performance with BERT. Moreover, they claimed that in automatic mental health systems, where labeled data is frequently scarce, recent transfer learning algorithms should become a crucial component.

Du et al. [13] looked at approaches for identifying suicide-related psychiatric stresses in Twitter data using deep learning-based approaches and a transfer learning approach that uses an existing clinical text annotation dataset. They demonstrated the advantages of deep learning-based techniques compared to conventional machine learning algorithms.

Additionally, it was discovered that the transfer learning technique might potentially reduce annotation work and further improve performance. To automatically detect public opinions, behavioral intentions, and attitudes concerning COVID-19 vaccinations from Tweets, Cagliero and Garza [29] used transfer learning with a pre-trained BERT model.

They showed that transfer learning models outperformed traditional machine learning models. To summarize, ML and DL techniques have been used in health care problems as efficient methods using text on social media platforms due to their ability to outperform naive learning models significantly [9].

Motivated by these models, we proposed a Transformer model with late fusion methods to combine the title and post of the dataset into the model to detect the mental disorders of individuals. To the best of our knowledge, none of the prior studies have applied the Transformer model with late fusion models for the mental illness problem.

3 Problem Description and Dataset

3.1 Mental Illness Problem

Mental illness problem is a multi-class classification problem where a given text is classified into one of the six following mental disorder classes:

– **ADHD**: A mental condition that impairs your ability to focus, maintain stillness, and control your actions (common in children)[3].

– **Anxiety**: A feeling of uneasiness, fear, and dread[4].

[3] www.cdc.gov/ncbddd/adhd/facts.html Last visited: 125-09-2023

[4] /medlineplus.gov/anxiety.html\#:~:text=Anxiety\%20is\%20a\%20feeling\%20of,before\%20making\%20an\%20important\%20decision Last visited: 25-09-2023

Table 5. Confusion matrix of the transformer model with concatenation late fusion method

		Predicted					
		ADHD	**Anxiety**	**Bipolar**	**Depression**	**PTSD**	**None**
True	**ADHD**	224	10	5	6	3	0
	Anxiety	1	222	1	19	5	0
	Bipolar	9	8	211	16	3	1
	Depression	3	8	12	219	6	0
	PTSD	0	19	9	7	213	0
	None	0	1	1	1	1	244

– **Bipolar**: Extreme mood swings, including emotional highs and lows, are a symptom of a mental health issue [5].

– **Depression**: A widespread and significant medical condition that has a negative impact on how someone feels, thinks, and acts[6].

– **PTSD**: A condition that some people experience after going through a stressful, terrifying, or deadly experience[7].

– **None**: No mental illness.

3.2 Dataset

Murarka et al. [35] developed a benchmark multi-class dataset from the Reddit social media platform for mental illness detection.

The dataset comprises a total of $16,703$ posts. The dataset was further divided into training, development, and test sets.

Table 2 presents the number of posts for each mental illness class.

[5] www.mayoclinic.org/diseases-conditions/bipolar-disorder/symptoms-causes/syc-20355955 Last visited: 25-09-2023.

[6] www.psychiatry.org/patients-families/depression/what-is-depression Last visited: 25-09-2023.

[7] www.nimh.nih.gov/health/topics/post-traumatic-stress-disorder\\-ptsd\#:~:text=Post\%2Dtraumatic\%20stress\%20disorder\%20(PTSD,danger\%20or\%20to\%20avoid\%20it. Last visited: 25-09-2023.

4 Model

4.1 Transformer Model

Transformer model [47] is gaining interest due to state-of-the-art performance in NLP tasks such as machine translation [48, 30], and sequence tagging [46, 20]. The Transformer model comprises encoder-decoder architectures that process sequential data in parallel without a recurrent network.

Instead of paying attention to the last state of the encoder, as is common with RNNs, the encoder architecture in Transformer extracts information from the whole sequence. This allows the decoder to assign greater weight to a certain input element for each output element.

In this study, we proposed Transformer models based on the vanilla Transformer proposed by Vaswani et al. [47] and used the encoder module of the Transformer to perform classification by mapping the data to the mental illness classes. The architecture of the Transformer model is shown in Figure 2.

Let $S = \{X_i, W_i, m_i\}_{i=1}^{T}$ denote a set of T samples, where X_i is a title, W_i is a post. m_i is the corresponding mental illness class (adhd, anxiety, bipolar, depression, ptsd, none). The words $\{w_1, w_2, \cdots, w_n\}$ for a text, which can be *title* or *post*, are mapped to the corresponding embeddings in the embedding layer, and the positional information E_{pos} is encoded and appended to the text representation and fed into the encoder layer, which consists L identical layers. The classification layer is a

Table 6. Results of the proposed model with baseline models

Model	**F1**	**Precision**	**Recall**
Transformer	**89.65**	**89.86**	**89.58**
Classical Machine Learning			
ML Algorithm	**F1**	**Precision**	**Recall**
LinearSVC	77.18	77.66	77.15
LR	77.87	78.24	77.89
NB	66.49	72.18	66.73
RF	70.85	72.46	70.50
Deep Learning			
DL Algorithm	**F1**	**Precision**	**Recall**
CNN	81.64	82.84	82.65
LSTM	83.73	84.10	83.60
BiLSTM	83.84	84.06	83.74
Transfer Learning			
TL Algorithm	**F1**	**Precision**	**Recall**
BERT	80.82	80.87	80.85
AlBERT	80.45	80.90	80.38
RoBERTa	84.41	85.10	84.41
State-of-the-Art			
Method	**F1**	**Precision**	**Recall**
RoBERTa	89	89	89

softmax layer that takes the average of the last transformer encoder layer o and multiplies the corresponding weights to get classification:

$$\hat{s} = \text{softmax}(W \times o + b), \tag{1}$$

where $\hat{s}$ is the predicted result through the model, W is the weighted matrix, and b is the bias.

How do title and post contribute to the predictions? Over the years, various fusion techniques (e.g., early fusion or late fusion) have been developed for prediction in computer vision [22, 18] and NLP tasks [45, 34].

Since there are two parts for each instance (title and post), we also applied late fusion combining the outputs of each model at the classification layer. Moreover, we tried various combinations of methods in experimental settings (e.g., concatenation, average, maximum, minimum, weighted average).

4.2 Baseline Models

Since the mental illness dataset used in this study is relatively new, we applied ML, DL, and TL algorithms to get baseline scores. The models are summarised as follows:

- **Machine Learning Classifiers:** We applied four different ML classifiers, including Random Forest, Linear Support Vector Machine, Multinomial Naive Bayes, and Logistic Regression using sckit-learn library[8].
- **Deep Learning Methods:** We applied base DL models: LSTM, BiLSTM, and CNN. The pre-trained embeddings were used as the input layer, and the softmax layer as the output layer of the models.
- **Transfer Learning Methods:** Transformer-based pre-trained language models (PLMs) such as BERT [11], RoBERTa [31], AlBERT [27] have shown state-of-art performance in many down-stream NLP tasks. The PLMs used in NLP problems, called transfer learning models, yielded top results in various NLP tasks without critical task-specific design changes [28, 11]. We employed the BERT, AlBERT, and RoBERTa models in this study.

5 Results and Analysis

5.1 Experimental Setting

We implemented the proposed DL and TL models using the PyTorch library [38]. The Adam optimizer [26] was used with an epsilon value of $1e-8$ and the default max grad norm. We used early stopping with 5 patience.

We utilized pre-trained language models (BERT [11], RoBERTa [31], etc.) to convert words into embeddings. To tokenize the words, we set the maximum length 35 and 512 for the title and post, respectively, for all pre-trained language models (BERT, RoBERTa, etc.).

[8] scikit-learn.org/ Last visited: 125-09-2023.

Table 7. Results of the proposed model with baseline models

	Title			Post		
Model	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**
Transformer	70.09	70.46	70.09	83.63	83.90	83.53
ML Algorithms	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**
Classical Machine Learning						
LinearSVC	65.48	65.78	65.52	77.18	77.66	77.15
LR	65.37	66	65.52	77.87	78.24	77.89
NB	62.46	68.56	62.37	66.49	72.18	66.73
RF	61.63	62.02	61.63	70.85	72.46	70.50
Deep Learning						
DL Algorithm	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**
CNN	69.94	70.85	69.76	81.64	82.84	82.65
LSTM	71.46	72.07	71.44	83.73	84.10	83.60
BiLSTM	70.12	70.65	69.89	83.84	84.06	83.74
Transfer Learning						
DL Algorithm	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**
BERT	70.06	70.29	69.96	80.82	80.87	80.85
AlBERT	67.37	67.58	67.34	80.45	80.90	80.38
RoBERTa	70.68	71.27	70.56	**84.41**	**85.10**	**84.41**

For TL models, we added an output layer with a softmax function for training and set the learning rate to $1e-5$ and the batch size to 16.

In ML models, the number of features in each experiment was set to 1,000, i.e., we used the n-grams with the highest TF-IDF values. For the combination of word n-grams, the length of N was minimum = 1 and maximum = 3.

Since the dataset was already pre-processed by eliminating URLs or usernames containing sensitive material, we did not apply any pre-processing techniques before classification. We fine-tuned the models using the development set of the dataset.

Table 3 shows the parameter settings of DL and the proposed transformer models. We evaluated the models using the following three metrics: micro precision, micro recall, and micro F1-Score.

5.2 Main Results

Table 6 presents the proposed models' results and comparison with baseline models. In this Table, "ML Algorithms" indicates traditional ML algorithms. The "LinearSVC" indicates Linear Support Vector Classifier, "LR" indicates Logistic Regression, "NB" indicates Naive Bayes, and "RF" indicates Random Forest classifier.

The "DL Algorithms" indicates DL algorithms used in this study, such as CNN, LSTM, and BiLSTM. The "TL Algorithms" refer to pre-trained TL algorithms applied to evaluate Reddit corpus, i.e., BERT, XLNet, AlBERT, and RoBERTa. Using traditional ML algorithms, overall, best results (F1 = 77.87) are obtained using a combination of word n-grams when the length of *N* was minimum = 1, maximum = 3 with the Logistic Regression model. This shows that combinations of word grams (length of *N* = 1-3) were the most suitable

Table 8. Class-wise results

	Title			Post			Title + Post		
Class	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**
ADHD	68	77	72	85	87	86	95	90	92
Anxiety	60	69	64	72	83	77	83	90	86
Bipolar	64	55	59	82	78	80	88	85	87
Depression	69	65	67	80	77	78	82	88	85
PTSD	67	65	66	86	83	84	82	88	85
None	95	90	92	98	94	96	82	88	85

features when we trained the model on the Reddit social media platform.

In DL models, the overall best results are achieved with BiLSTM (F1 = 83.84), which shows that DL models are suitable for detecting mental illness. Additionally, DL results are almost similar to the results of TL models.

Since RoBERTa pre-trained model in TL methods yielded the best results, we used RoBERTa pre-trained embeddings as the input layer of the DL models (CNN, LSTM, and BiLSTM) and the proposed Transformer model.

The state-of-the-art RoBERTa [35] model was trained on title + post text, which is different from our RoBERTa model as we trained it on posts only. Among the baseline models (ML, DL, and TL), RoBERTa outperformed the traditional ML and DL models with an F1 score of 84.41 on this challenging multi-class mental illness detection problem.

Overall, we obtained the highest score with the proposed Transformer model with the `concatenation` late fusion method (F1 = 89.65). Our proposed model outperformed the state-of-the-art RoBERTa [35] model (F1 = 89).

Table 5 shows the confusion matrix of the proposed Transformer model with the concatenation late fusion method. The model is good at predicting non-illness samples. However, it confuses at prediction of the classes anxiety, bipolar, depression, and ptsd.

The terms `Depression` and `Anxiety` are presented in data instances of the `ADHD` and `PTSD` classes more than these class themselves. One could expect poor outcomes due to this, but these classes outperformed all others.

This exhibits the actual potential of our approach since it does not depend solely on the mention of class names in the post but also has a deep awareness of the post's context.

5.3 Data

To understand the impact of the dataset comprising titles and posts, we performed experiments with the proposed Transformer and the baseline models using title and post separately. Table 7 represented the F1, Precision, and Recall scores of Transformer, traditional ML, and DL models.

We obtained the best results with the posts (F1 = 84.41). We analyzed that the length of the titles in the dataset is shorter than the posts, which indicates that they are not informative enough. Therefore, we can say that the length of the text is important for the models, especially for the DL.

To understand the impact of the methods, Table 8 presented the class-wise results of the Transformer using the title and post separately and concatenating them.

We performed experiments using the proposed Transformer model to get insights on the class-wise performance of our proposed Transformer model on titles and post text separately and by combining them. The Transformer model obtained a 0.96 F1 score for `none` class on posts only, which shows that this proposed model will suffer from minimum false positives in detecting mental illness on social media text. The Transformer model using title and post together increased the F1 score of each

Table 9. Class-wise results of late fusion methods

		ADHD	Anxiety	Bipolar	Depression	PTSD	None
Concatenation	Precision	95	83	88	82	92	100
	Recall	90	90	85	88	86	98
	F1	92	86	87	85	89	99
Average	Precision	94	82	86	78	92	98
	Recall	90	87	83	86	82	97
	F1	92	85	84	82	87	97
Weighted Average	Precision	94	69	88	83	88	98
	Recall	84	91	82	77	84	97
	F1	89	79	85	80	86	97
Maximum	Precision	94	81	81	80	90	100
	Recall	89	86	86	83	84	97
	F1	92	84	84	82	87	98
Minimum	Precision	92	81	87	81	89	98
	Recall	92	86	81	88	82	98
	F1	92	84	84	84	86	98

mental illness class. However, the performance of `none` class decreased compared to other models (Transformer for title and post separately).

The best performing class among the mental disorders was `adhd`, while the performance of the other classes was similar. This shows that the model significantly fits the dataset. The training dataset contains fewer samples of `ptsd` class, and despite this, the F1 score of `ptsd` class was not dropped. The class-wise results were very similar to the RoBERTa [35] model except for the `none` class. Their model performed well on `none` class.

5.4 Late Fusion

To extend the impact of the data on the problem, we applied late fusion (Figure 1). The results of the methods used in late fusion are shown in Table 4. We used RoBERTa [31] pre-trained embeddings in the models with the same parameters for each model (See Table 3 for hyperparameter settings of the models).

The results showed that all methods improved the results compared to the Transformer model using only one input (title or post). We achieved the highest score (F1 = 89.65) by *concatenation* fusion methods. Table 9 presents the class-wise results of the Transformer model with late fusion methods.

It can be observed that the late fusion method of concatenation performed better on all classes than other methods. Moreover, there is not much difference in the performances of the late fusion methods. It can be inferred that the method can be used for datasets containing two or more texts to increase performance.

6 Conclusion

The present Covid-19 outbreak and globally forced isolation were our primary motivations for multi-class mental illness detection efforts. We believe that social media platforms have become the most widely used communication medium for individuals, allowing them to express themselves without fear of judgment.

We applied the Transformer model with fusion methods and state-of-the-art traditional ML, DL, and TL-based methods for multi-class mental illness detection problem. The best results (see

Table 4) were obtained with the Transformer model with concatenation late fusion method (F1 score = 89.65).

In the future, we plan to develop a multi-label mental illness dataset, which would be more reflective of the situation than a multi-class dataset, as a post can have more than one mental disease instead of one per post, i.e., depression and anxiety.

We can also use the data augmentation technique on top of existing mental health data [35]. Moreover, we plan to apply other TL-based models, such as DistilBERT, in the future. An ensemble modeling would also be considered to improve classification performance.

Acknowledgments

The work was done with partial support from the Mexican Government through the grant A1-S-47854 of CONACYT, Mexico, grants 20241816, 20241819, and 20240951 of the Secretaría de Investigación y Posgrado of the Instituto Politécnico Nacional, Mexico. The authors thank the CONACYT for the computing resources brought to them through the Plataforma de Aprendizaje Profundo para Tecnologías del Lenguaje of the Laboratorio de Supercómputo of the INAOE, Mexico, and acknowledge the support of Microsoft through the Microsoft Latin America PhD Award.

References

1. **Abusaa, M., Diederich, J., Al-Ajmi, A. (2004).** Machine learning, text classification and mental health. 12th National Health Informatics Conference, pp. 1–7.
2. **Ameer, I., Ashraf, N., Sidorov, G., Gómez-Adorno, H. (2020).** Multi-label emotion classification using content-based features in twitter. Computación y Sistemas, Vol. 24, No. 3, pp. 1159–1164.
3. **Ameer, I., Siddiqui, M. H. F., Sidorov, G., Gelbukh, A. (2019).** CIC at SemEval-2019 task 5: Simple yet efficient approach to hate speech detection, aggressive behavior detection, and target classification in Twitter. Proceedings of the 13th International Workshop on Semantic Evaluation, pp. 382–386. DOI: 10.18653/v1/S19-2067.
4. **Ameer, I., Sidorov, G. (2021).** Author profiling using texts in social networks. In Handbook of Research on Natural Language Processing and Smart Service Systems. IGI Global, pp. 245–265. DOI: 10.4018/978-1-7998-4730-4.ch011.
5. **Ameer, I., Sidorov, G., Gomez-Adorno, H., Nawab, R. M. A. (2022).** Multi-label emotion classification on code-mixed text: Data and methods. IEEE Access, Vol. 10, pp. 8779–8789. DOI: 10.1109/ACCESS.2022.3143819.
6. **Ameer, I., Sidorov, G., Nawab, R. M. A. (2019).** Author profiling for age and gender using combinations of features of various types. Journal of Intelligent & Fuzzy Systems, Vol. 36, No. 5, pp. 4833–4843. DOI: 10.3233/JIFS-179031.
7. **Benton, A., Mitchell, M., Hovy, D. (2017).** Multi-task learning for mental health using social media text. CoRR. DOI: 10.48550/arXiv.1712.03538.
8. **Biship, C. M. (2007).** Pattern recognition and machine learning. Springer New York, NY.
9. **Cagliero, L., Garza, P. (2013).** Improving classification models with taxonomy information. Data & Knowledge Engineering, Vol. 86, pp. 85–101. DOI: 10.1016/j.datak.2013.01.005.
10. **Coppersmith, G., Dredze, M., Harman, C., Hollingshead, K., Mitchell, M. (2015).** CLPsych 2015 shared task: Depression and PTSD on Twitter. Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality, pp. 31–39.

11. **Devlin, J., Chang, M., Lee, K., Toutanova, K. (2018).** BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies 2019, pp. 4171–4186. DOI: 10.48550/arXiv.1810.04805.
12. **Dhanalaxmi, S., Agarwal, R., Sinha, A. (2020).** Detection of COVID-19 informative tweets using RoBERTa. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing Workshop W-NUT: The Sixth Workshop on Noisy User-generated Text, pp. 409–413. DOI: 10.48550/arXiv.2010.11238.
13. **Du, J., Zhang, Y., Luo, J., Jia, Y., Wei, Q., Tao, C., Xu, H. (2018).** Extracting psychiatric stressors for suicide from social media using deep learning. BMC Medical Informatics and Decision Making, Vol. 18, No. 43, pp. 77–87. DOI: 10.1186/s12911-018-0632-8.
14. **Durstewitz, D., Koppe, G., Meyer-Lindenberg, A. (2019).** Deep neural networks in psychiatry. Molecular Psychiatry, Vol. 24, No. 11, pp. 1583–1598. DOI: 10.1038/s41380-019-0365-9.
15. **Dwyer, D. B., Falkai, P., Koutsouleris, N. (2018).** Machine learning approaches for clinical psychology and psychiatry. Annual review of clinical psychology, Vol. 14, pp. 91–118. DOI: 10.1146/annurev-clinpsy-032816-045037.
16. **Gillioz, A., Casas, J., Mugellini, E., Abou-Khaled, O. (2020).** Overview of the transformer-based models for NLP tasks. 2020 15th Conference on Computer Science and Information Systems (FedCSIS), pp. 179–183. DOI: 10.15439/2020F20.
17. **Gkotsis, G., Oellrich, A., Velupillai, S., Liakata, M., Hubbard, T. J., Dobson, R. J., Dutta, R. (2017).** Characterisation of mental health conditions in social media using informed deep learning. Scientific Reports, Vol. 7, No. 1, pp. 1–11. DOI: 10.1038/srep45141.
18. **Gunes, H., Piccardi, M. (2005).** Affect recognition from face and body: Early fusion vs. late fusion. 2005 IEEE international conference on systems, man and cybernetics, IEEE, Vol. 4, pp. 3437–3443. DOI: 10.1109/ICSMC.2005.1571679.
19. **Hamilton, M. (1967).** Development of a rating scale for primary depressive illness. British journal of social and clinical psychology, Vol. 6, No. 4, pp. 278–296. DOI: 10.1111/j.2044-8260.1967.tb00530.x.
20. **He, Z., Wang, Z., Wei, W., Feng, S., Mao, X., Jiang, S. A. (2020).** A survey on recent advances in sequence labeling from deep learning models. ArXiv, Vol. abs/2011.06727. DOI: 10.48550/arXiv.2011.06727.
21. **Hu, Y., Sokolova, M. (2021).** Explainable multi-class classification of the CAMH COVID-19 mental health data. ArXiv. DOI: 10.48550/arXiv.2105.13430.
22. **Ionescu, B., Benois-Pineau, J., Piatrik, T., Quénot, G. (2014).** Fusion in computer vision: Understanding complex visual content. Springer. DOI: 10.1007/978-3-319-05696-8.
23. **Ive, J., Viani, N., Kam, J., Yin, L., Verma, S., Puntis, S., Cardinal, R. N., Roberts, A., Stewart, R., Velupillai, S. (2020).** Generation and evaluation of artificial mental health records for natural language processing. NPJ Digital Medicine, Vol. 3, No. 1, pp. 1–9. DOI: 10.1038/s41746-020-0267-x.
24. **Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., Shah, M. (2021).** Transformers in vision: A survey. ACM Computing Surveys (CSUR), Vol. 54, No. 10s, pp. 1–41. DOI: 10.1145/3505244.
25. **Kim, J., Lee, J., Park, E., Han, J. (2020).** A deep learning model for detecting mental illness from user content on social media. Scientific Reports, Vol. 10, No. 1, pp. 11846. DOI: 10.1038/s41598-020-68764-y.
26. **Kingma, D. P., Ba, J. (2014).** Adam: A method for stochastic optimization. International Conference on Learning Representations. DOI: 10.48550/ARXIV.1412.6980.

27. **Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R. (2019).** Albert: A lite bert for self-supervised learning of language representations. International Conference on Learning Representations.

28. **Li, X., Fu, X., Xu, G., Yang, Y., Wang, J., Jin, L., Liu, Q., Xiang, T. (2020).** Enhancing BERT representation with context-aware embedding for aspect-based sentiment analysis. IEEE Access, Vol. 8, pp. 46868–46876. DOI: 10.1109/ACCESS.2020.2978511.

29. **Liu, S., Li, J., Liu, J. (2021).** Leveraging transfer learning to analyze opinions, attitudes, and behavioral intentions toward COVID-19 vaccines: Social media content and temporal analysis. Journal of Medical Internet Research, Vol. 23, No. 8, pp. e30251. DOI: 10.2196/30251.

30. **Liu, X., Duh, K., Liu, L., Gao, J. (2020).** Very deep transformers for neural machine translation. ArXiv. DOI: 0.48550/arXiv.2008.07772.

31. **Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V. (2019).** Roberta: A robustly optimized BERT pretraining approach. The International Conference on Learning Representations 2020 Conference Blind Submission. DOI: 10.48550/arXiv.1907.11692.

32. **Marcus, M., Yasamy, M. T., van-Ommeren, M., Chisholm, D., Saxena, S. (2012).** Depression: A global public health concern. DOI: 10.1037/e517532013-004.

33. **Mathur, P., Sawhney, R., Chopra, S., Leekha, M., Shah, R. R. (2020).** Utilizing temporal psycholinguistic cues for suicidal intent estimation. Advances in Information Retrieval, Springer International Publishing, Vol. 12036, pp. 265–271. DOI: 10.1007/978-3-030-45442-5_33.

34. **Mukherjee, S. (2019).** Deep learning technique for sentiment analysis of hindi-english code-mixed text using late fusion of character and word features. 2019 IEEE 16th India Council International Conference (INDICON), pp. 1–4. DOI: 10.1109/INDICON47234.2019.9028928.

35. **Murarka, A., Radhakrishnan, B., Ravichandran, S. (2021).** Classification of mental illnesses on social media using RoBERTa. Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis, pp. 59–68.

36. **Murphy, K. P. (2012).** Machine learning: A probabilistic perspective. MIT press, Cambridge, Massachusetts, USA.

37. **Orabi, A. H., Buddhitha, P., Orabi, M. H., Inkpen, D. (2018).** Deep learning for depression detection of twitter users. Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic, pp. 88–97. DOI: 10.18653/v1/W18-0609.

38. **Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S. (2019).** Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, Vol. 32.

39. **Pervaz, I., Ameer, I., Sittar, A., Nawab, R. M. A. (2015).** Identification of author personality traits using stylistic features: Notebook for pan at clef 2015. CLEF (Working Notes), pp. 1–7.

40. **Ribeiro, M. T., Singh, S., Guestrin, C. (2016).** " Why should i trust you?" Explaining the predictions of any classifier. Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135–1144. DOI: 10.1145/2939672.2939778.

41. **Sekulic, I., Strube, M. (2020).** Adapting deep learning methods for mental health prediction on social media. Proceedings of the 5th Workshop on Noisy User-generated Text

(W-NUT 2019), pp. 322–327. DOI: 10.18653/v1/D19-5542.

42. **Shatte, A. B., Hutchinson, D. M., Teague, S. J. (2019).** Machine learning in mental health: a scoping review of methods and applications. Psychological medicine, Vol. 49, No. 9, pp. 1426–1448. DOI: 10.1017/S0033291719000151.

43. **Shickel, B., Heesacker, M., Benton, S., Rashidi, P. (2020).** Automated emotional valence prediction in mental health text via deep transfer learning. 2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE), pp. 269–274. DOI: 10.1109/BIBE50027.2020.00051.

44. **Sittar, A., Ameer, I. (2018).** Multi-lingual author profiling using stylistic features. FIRE (Working Notes), pp. 240–246.

45. **Soriano-Morales, E. P., Ah-Pine, J., Loudcher, S. (2017).** Fusion techniques for named entity recognition and word sense induction and disambiguation. Discovery Science: 20th International Conference, Springer International Publishing, pp. 340–355.

46. **Tsai, H., Riesa, J., Johnson, M., Arivazhagan, N., Li, X., Archer, A. (2019).** Small and practical BERT models for sequence labeling. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3632–3636. DOI: 10.18653/v1/D19-1374.

47. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2017).** Attention is all you need. Advances in Neural Information Processing Systems, Vol. 30, pp. 5998–6008.

48. **Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D. F., Chao, L. S. (2019).** Learning deep transformer models for machine translation. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1810–1822. DOI: 10.18653/v1/P19-1176.

49. **Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von-Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le-Scao, T., Gugger, S., Drame, M., et al. (2020).** Transformers: State-of-the-art natural language processing. Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6.

50. **World Health Organization (2001).** The world health report 2001: Mental health: New understanding, new hope. https://iris.who.int/handle/10665/42390.

51. **Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Hou, Q., Feng, J. (2021).** DeepViT: Towards deeper vision transformer. arXiv. DOI: 10.48550/arXiv.2103.11886.

52. **Zirikly, A., Resnik, P., Uzuner, O., Hollingshead, K. (2019).** CLPsych 2019 shared task: Predicting the degree of suicide risk in Reddit posts. Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology, pp. 24–33. DOI: 10.18653/v1/W19-3003.

Article received on 26/02/2024; accepted on 12/04/2024.
**Corresponding author is Grigori Sidorov.*

ISSN 2007-9737

An Adaptative Eps Parameter of DBSCAN Algorithm for Identifying Clusters with Heterogeneous Density

Nasereddine Amroune[1,2,*], Maklouf Benazi[1,2], Lamri Sayad[1,2]

[1] Mohamed Boudiaf University of M´sila,
Department of computer science, M´sila,
Algeria

[2] University of M'sila,
Laboratory of Informatics and its Applications of M´sila,
Algeria

{nasereddine.amroune, maklouf.benazi, lamri.sayad}@univ-msila.dz

Abstract. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is one of the most important data clustering algorithms. Its importance lies in the fact that it can recognize clusters of arbitrary shapes and is not affected by noise in the data. To identify clusters, DBSCAN needs to specify two parameters: the parameter Eps, representing the radius of the circle to identify the neighborhood of each observation. The second parameter of DBSCAN is minpts, which represents the minimum size of the neighborhood for a point to be a seed in a cluster and not a noise. However, the task of determining the adequate value of Eps parameter is not easy and represents a major issue when applying DBSCAN since the accuracy of this algorithm highly depends on the values of its parameters. In this paper, we present a new version of DBSCAN where we need only to specify the minpts parameter, then we use k-nearest neighbors (kNN) algorithm to calculate the value of Eps automatically for every point in the data. This technique not only reduces the number of parameters by eliminating Eps which has been very difficult to determine, but also gives DBSCAN the ability to detect clusters with heterogeneous density. The experimental results show that the proposed method is more efficient and more accurate than the original DBSCAN algorithm.

Keywords. Data mining, clustering, density-based algorithms, DBSCAN algorithm, eps parameter, k-nearest neighbors.

1 Introduction

Over the last decades, unsupervised classification, or more precisely data clustering, has become one of the most important techniques in machine learning. This technique allows to identify significant patterns in the dataset based on their similarity in order to better understand them and make more accurate predictions.

It consists in grouping unlabeled objects of a dataset in subgroups called clusters. objects in each cluster should be as similar as possible to each other and as dissimilar as possible to members of other clusters.

The clustering technique has applications in many areas of the real world. for example, astronomy [1], biology [2], engineering [3], computer science [4], marketing [5], insurance [6], medicine [7], etc. In the literature, many clustering methods and algorithms have been proposed. They can be grouped into four categories.

– Partitioning methods where objects are divided into k distinct groups so that the objects in each group are as close as possible to the center of its group and as far away as possible from the centers of the other groups.
– The typical examples of this category are k-means [8] and k-median [9].

- Hierarchical methods are types of approaches where we start with one element in each cluster, then we merge iteratively the two most similar clusters until only one is left like in SLINK [10], or conversely, we start with one cluster and split it until each element has its own cluster like in CLINK [11]. In this type of clustering, it is necessary to provide another method or way to determine at what level to stop the merge or división process.
- Model-based methods: In this category, the cluster is seen as a distribution in a mixture of distributions (e.g., gaussian mixture) and the objective of clustering is to find the original models that generate these distributions. such as the work proposed in [12].
- Density-based methods: in this type of clustering, the cluster is seen as a set of high-density objects separated by low-density objects.
- Algorithms of this type can easily identify clusters of arbitrary shape as they can also handle noise and outliers in the data. The two best known algorithms in this family are DBSCAN [13] and OPTICS [14].

DBSCAN Density-Based Spatial Clustering of Applications with Noise is a density-based clustering algorithm that can identify clusters of arbitrary shapes and is not affected by noise in the data.

However, DBSCAN has two drawbacks: to identify clusters, it needs two parameters, Eps and minpts, which are very difficult to specify, and its inability to detect clusters with heterogeneous density. In this article, we present a new version of DBSCAN which can identify clusters of arbitrary shapes with heterogeneous density, and which only requires the specification of the minpts parameter. the algorithm then uses the k-nearest neighbors (kNN) algorithm to automatically calculate Eps for all points in the data.

The rest of this paper is organized as follows: In section 2, we present the basic concept of DBSCAN algorithm. Then, some variants of DBSCAN algorithm are explained in section 3. Section 4 is devoted to presenting our approach in detail. To show the performance of our method, section 5 gives some results of the application of our method on different datasets well known in the clustering world. Finally, section 6 concludes the paper and gives some perspectives.

2 DBSCAN Algorithm

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm, that can identify clusters with arbitrary shapes and different sizes. Unlike k-means, clusters are not necessarily spheroidal and there is no need to know the number of clusters in advance.

Algorithm 1 DBSCAN algorithm

Input: X: dataset,
minpts: minimum points,
Eps: maximum distance
Output: Labels
1. Find sets of points within the epsilon neighborhood of each point in X.
2. Select the first unclassified point
  a. If the number of neighbors is less than minpts, then label the point as a noise point. Go to step 2.
  b. Otherwise, label the current point as a core point belonging to cluster C. and insert all neighbors into queue.
3. Iterate over all unclassified points in the queue
  a. If the current point is labeled as noise or the number of neighbors is less than minpts, than label the current point as a border point belonging to C cluster
  b. Otherwise, label the current point as a core point belonging to cluster C. and add all its neighbors into the queue.
4. Select the next unclassified point in X, and increase the cluster count by 1.
5. Repeat steps 2–4 until all points in X are labeled.

Clusters are formed by the identifying points that are density-connected. The algorithm uses two parameters, minpts and eps, to define density in data. It can identify three types of points: core points, border points, and noise points [13], which make it more suitable for processing noisy. To assign a point to a cluster, the point must have at least minpts in its radius neighborhood (eps), in this case, we call it core point.

Or, the point is within the radius neighborhood of another core point, in this case, we call it border point. Otherwise, it will be considered a noise point. The original DBSCAN algorithm is presented below.

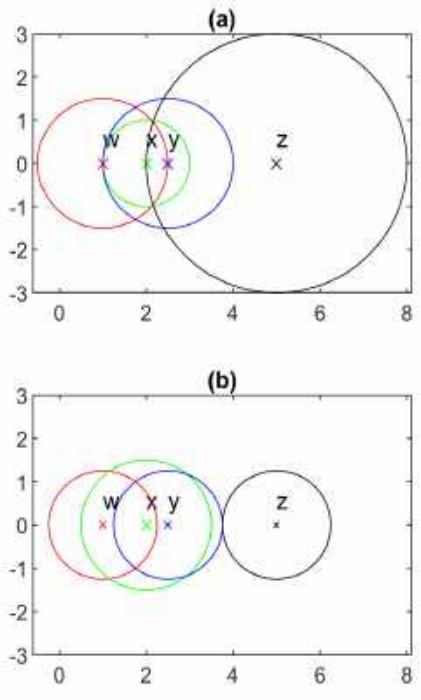


Fig. 1. Example to illustrate the results obtained by applying formula 1

To identify clusters, DBSCAN depends heavily on the parameter eps, which represents the maximum distance to search for neighbors.

In addition to being difficult to guess, using a global value for eps can lead DBSCAN to not find the right clusters, especially when data have clusters with different densities.

When taking eps with small values, low density clusters may be considered as noise. However, if eps take a large value, DBSCAN will merge high density clusters.

3 Related Work

Recent research has been focused on using a single parameter with the widely used clustering algorithm, DBSCAN, to reduce the complexity associated with parameter tuning while maintaining high-quality clustering results.

Various papers have proposed different approaches for using a single parameter with DBSCAN, and this approach has gained attention from the research community. In this section, the recent papers that explore these approaches and their potential to achieve high-quality clustering results will be reviewed and discussed.

A. Bryant and K. Cios [15] have developed a new clustering algorithm called RNN-DBSCAN, which estimates observation density using reverse nearest neighbor counts and employs a DBSCAN-like approach based on k-nearest neighbor graph traversals through dense observations.

This algorithm offers two significant benefits: it reduces problem complexity to a single parameter and enhances the ability to handle large variations in cluster density. However, it may not be effective in distinguishing adjacent clusters with different densities, which could limit its usefulness in some situations.

Hu et al. [16] proposed a new density-based clustering algorithm called KR-DBSCAN. The algorithm is based on the reverse nearest neighbor and influence space. KR-DBSCAN distinguishes adjacent clusters with different densities, reduces computational load, and identifies boundary objects and noise objects.

The algorithm defines a new cluster expansion condition using the reverse nearest neighborhood and its influence space and adds core objects to the cluster by breadth-first traversal.

However, KR-DBSCAN is more complex than standard DBSCAN due to the use of influence space, which can lead to higher computational cost for large databases. In addition to these approaches, other papers have proposed novel mechanisms to find the best value of epsilon for a given dataset.

For instance, the Multi-verse optimizer MVO algorithm [17] was used to find the Eps interval corresponding to the highest accuracy of DBSCAN. The AE-DBSCAN algorithm [18] uses the first slope, which is greater than the mean plus standard deviation of all non-zero slopes, as Eps.

AutoEpsDBSCAN [19] uses the kdist graph to select several values of the Eps parameter for different densities in the dataset before applying traditional DBSCAN. Overall, the research reviewed in this section indicates that using a single parameter with DBSCAN can be a promising alternative to the traditional two-parameter approach for DBSCAN.

Different approaches have been proposed, each with its strengths and limitations, and the

choice of the appropriate approach depends on the characteristics of the dataset and the desired clustering quality and efficiency.

4 Modified DBSCAN

In this section, we explain our proposed algorithm, which can be carried out in two steps. In the first step, the eps values are automatically computed for every point in the data. To do this, we calculate the k-distances of all its k nearest points and then, we calculate the average according to the following formula:

$$eps(i) = \frac{1}{k} \sum kdist(j) \quad \big(j \in N(i)\big), \tag{1}$$

where i is the point, whose eps will be calculated, k is the number of points in a neighborhood of i, in DBSCAN algorithm k is called minpts, N(i) is the set of nearest neighbors of the point i, and kdist is the function that calculates the distance between a point j and its k^{th} nearest neighbor.

Algorithm 2 Computing eps and set of neighbors

Input: X: Dataset, k: minimum points.
Output: eps,N:set of neighbors of each point in X
Calculate k-distance values of all points in X
for each observation x_i in X
Calculate eps(x_i),theaverage k-distance of k nearest neighbors of x_i.
Find N(x_i), set of neighbors of x_i within eps(x_i)
Mark x_i as unclassified
end for

The second step is to use DBSCAN to find clusters using eps computed in the first step.For a better understanding, the following example illustrates the results obtained by applying formula 1 to a simple mono-dimensional dataset of four points.

Fig 1.a shows the circles corresponding to the kth nearest neighbors for each point, here k=3, and Fig 1.b shows circles whose radius are equal to the average k-distance of their k nearest neighbors (formula 1).

We can see that radius of the point "x" is extended to be core point, while the radius of the other three points have been reduced to get fewer points in their neighborhoods (less than k), making the point "w" and the point "y" borders points because they fall within the neighborhood of "x" and make "z" as noise point.

Algorithm 3 Our DBSCAN algorithm

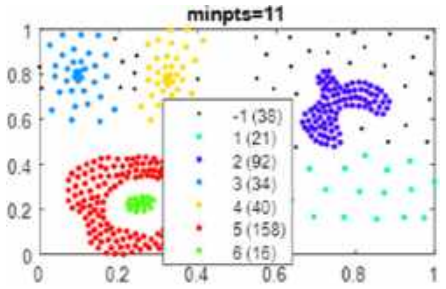
Input: X: Dataset,
N: Set of neighbors,
minpts: minimum points.
Output: Labels
$C = 0$
for each observation x_i in X
Select the next unclassified point in X
if $|N(xi)| <$ minpts
Mark x_i as noise.
Go to the next observation in X
end if
$C = C + 1$ // start a new cluster
Mark x_i as a core point belonging to cluster C
Insert $N(x_i)$, neighbors of x_i, into queue Q
while Q is not empty
Select qi the next unclassified point in Q
if q_i is marked as noise or $|N(q_i)| <$ minpts
Mark q_i as a border point belonging to cluster C.
else
Mark q_i as a core point belonging to cluster C.
Insert $N(q_i)$, neighbors of q_i, into queue Q.
end if
end while
end for

5 Experimental Evaluation

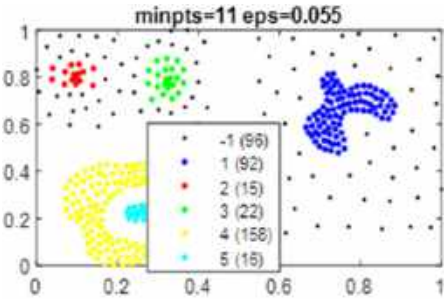
In the previous section, we presented a novel version of the DBSCAN algorithm that utilizes only the 'minPts' parameter for clustering.

In this section, we aim to provide empirical evidence of the effectiveness of our new algorithm by conducting a comparative analysis with the original DBSCAN algorithm, which uses two parameters.

To assess the effectiveness of our proposed algorithm, we utilized three synthetic 2-D datasets of different shapes and different densities. The first dataset, named 'Compound', comprises 399 points divided into six clusters with varying densities,

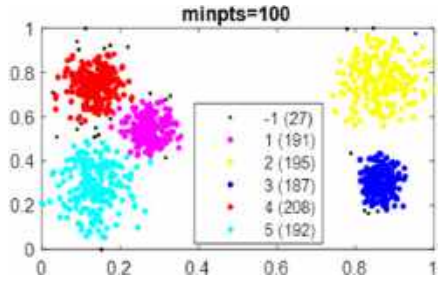


(a)

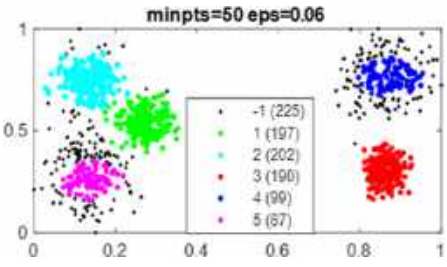


(b)

Fig. 2. Our-DBSCAN (a) and DBSCAN (b) clustering results for the compound dataset



(a)



(b)

Fig. 3. Our-DBSCAN (a) and DBSCAN (b) clustering results for the asymmetric dataset

posing a challenge to accurate clustering. The second dataset, 'asymmetric', is composed of 1000 points divided into five clusters.

The third dataset, 'hard', includes 1500 points divided into three clusters of different densities.

Figure 2 displays the clustering results of the first dataset, where our method successfully identified 6 distinct clusters with only 38 data points labeled as noise. In contrast, the original DBSCAN algorithm detected only 5 clusters and classified the sixth cluster as noise, resulting in 96 noise points.

Figure 3 illustrates the clustering results obtained by two algorithms. our algorithm successfully identified five distinct clusters with 27 points labeled as noise. On the other hand, the original DBSCAN algorithm could barely detect five clusters with 225 points marked as noise.

If we increase the value of epsilon, it will merge the two groups in the upper left, while decreasing the value of epsilon will classify the group in the upper right as noise.

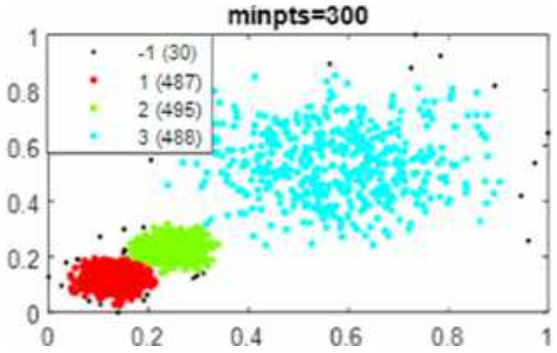
In reference to the hard database shown in Figure 4, it is not possible to determine a suitable value for the parameter eps that would allow us to identify the three clusters using the original DBSCAN algorithm.

This is because the algorithm relies on a fixed global value for eps, whereas our modified algorithm was able to successfully detect the three distinct clusters with only 30 points labeled as noise.

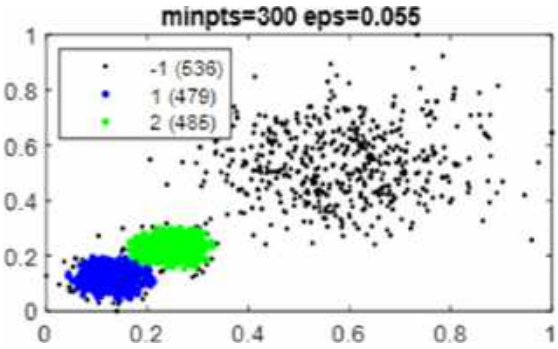
To further assess the effectiveness of our approach, we conducted experiments on widely-used datasets such as iris and seed. Since these datasets come with true labels, we were able to measure the accuracy of our method and compare it with that of the original DBSCAN algorithm.

Table 1 provides a summary of the results we obtained. As per the data presented in Table 1, it is evident that the proposed method exhibits better performance than the traditional DBSCAN algorithm on the iris and seed datasets. For the iris dataset, our DBSCAN algorithm achieved an accuracy of 0.86 compared to 0.67 with the traditional DBSCAN algorithm.

Additionally, our algorithm was able to detect three clusters with 17 noise points compared to two clusters with no noise points in the traditional algorithm.



(a)



(b)

Fig. 4. Our-DBSCAN (a) and DBSCAN (b) clustering results for the Asymmetric dataset

Table 1. Performance of our method on Real-World datasets

Dataset	Algorithm	Iris	Seed
DBSCAN	Accuracy	0.6667	0.5905
	NMI	0.7612	0.5536
	Cluster/Noise	2/0	3/81
	Minpts/eps	12/0.025	14/0.19
Our DBSCAN	Accuracy	0.8600	0.8048
	NMI	0.7376	0.6363
	Cluster/noise	3/17	5/15
	Minpts	12	14

This indicates that our algorithm is better at detecting the underlying structure in the iris dataset and is able to handle noisy data more effectively.

Similarly, for the seed dataset, our algorithm achieved an accuracy of 0.8048 compared to 0.5905 with the traditional DBSCAN algorithm. Our algorithm was also able to detect five clusters with only 15 noise points compared to three clusters with 81 noise points in the traditional algorithm.

This again shows that our algorithm is more robust and accurate in detecting the underlying structure in the data and is more effective in dealing with noisy data. Furthermore, our algorithm achieved a higher accuracy on both datasets even though the traditional algorithm achieved higher NMI on the iris dataset.

This suggests that our algorithm is more effective at finding meaningful clusters rather than just maximizing the agreement with the ground truth. Overall, the results suggest that the proposed DBSCAN algorithm is superior to the traditional DBSCAN algorithm in terms of accuracy, the number of clusters detected, and handling noisy data.

6 Conclusion

DBSCAN is a density-based clustering algorithm that can find clusters of arbitrary shapes and sizes. To discover the clusters, the algorithm depends on two parameters, minpts and eps. The problem with eps is that it is difficult to determine its value and takes a single global value for all data, which does not allow DBSCAN to discover clusters of different densities.

To solve these two problems, we proposed in this paper a new version of DBSCAN that can determine the eps value automatically and locally for each observation in the data. Experimental results show that this new version of DBSCAN can help identify good quality clusters.

As a perspective, we will try to figure out how to automatically determine the minpts value from the data, so that the algorithm will be completely parameter free.

References

1. **Yu, H., Hou, X. (2022).** Hierarchical clustering in astronomy. Astronomy and Computing, Vol. 41, pp. 100662. DOI: 10.1016/j.ascom.2022.100662.

2. **Ye, X., Ho, J. W. K. (2019).** Ultrafast clustering of single-cell flow cytometry data using

flowgrid. BMC Systems Biology, Vol. 13, No. 35. DOI: 10.1186/s12918-019-0690-2.

3. **Pham, D. T., Afify, A. A. (2007).** Clustering techniques and their applications in engineering. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, Vol. 221, No. 11, pp. 1445–1459. DOI: 10.1243/09544 062jmes508.
4. **Lu, X., Zhou, Y., Qiao, L., Yu, W., Liang, S., Zhao, M., Zhao, Y., Lu, C., Chi, N. (2019).** Amplitude jitter compensation of PMA-8 VLC system employing time-amplitude two-dimensional re-estimation base on density clustering of machine learning. Physica Scripta, Vol. 94, No. 5, pp. 055506. DOI: 10.10 88/1402-4896/ab0a9f.
5. **Ližbetinová, L., Štarchoň, P., Lorincová, S., Weberová, D., Průša, P. (2019).** Application of cluster analysis in marketing communications in small and medium-sized enterprises: an empirical study in the Slovak Republic. Sustainability, Vol. 11, No. 8, pp. 2302. DOI: 10.3390/su11082302.
6. **Gramegna, A., Giudici, P. (2020).** Why to buy insurance? an explainable artificial intelligence approach. Risks, Vol. 8, No. 4, pp. 137.DOI: 10.3390/risks8040137.
7. **Zhang, X., Wang, D., Chen, H. (2019).** Improved biogeography-based optimization algorithm and its application to clustering optimization and medical image segmentation. IEEE Access, Vol. 7, pp. 28810–28825. DOI: 10.1109/access.2019. 2901849.
8. **Likas, A., Vlassis, N., Verbeek, J. J. (2003).** The global k-means clustering algorithm. Pattern Recognition, Vol. 36, No. 2, pp. 451–461. DOI: 10.1016/s0031-3203(02) 00060-2.
9. **Kohonen, T. (1985).** Median strings. Pattern Recognition Letters, Vol. 3, No. 5, pp. 309–313. DOI: 10.1016/0167-8655(85)9 0061-3.
10. **Müllner, D. (2011).** Modern hierarchical, agglomerative clustering algorithms. DOI: 10. 48550/ARXIV.1109.2378.
11. **Defays, D. (1977).** An efficient algorithm for a complete link method. The Computer Journal, Vol. 20, No. 4, pp. 364–366. DOI: 10.1093/com jnl/20.4.364.
12. **Yousri, N. A., Kamel, M. S., Ismail, M. A. (2009).** A distance-relatedness dynamic model for clustering high dimensional data of arbitrary shapes and densities. Pattern Recognition, Vol. 42, No. 7, pp. 1193–1209. DOI: 10.1016/j. patcog.2008.08.037.
13. **Ester, M., Kriegel, H., Sander, J., Xu, X. (1996).** A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231.
14. **Ankerst, M., Breunig, M. M., Kriegel, H., Sander, J. (1999).** Optics: ordering points to identify the clustering structure. ACM SIGMOD Record, Vol. 28, No. 2, pp. 49–60. DOI: 10.11 45/304181.304187.
15. **Bryant, A., Cios, K. (2018).** RNN-DBSCAN: A density-based clustering algorithm using reverse nearest neighbor density estimates. IEEE Transactions on Knowledge and Data Engineering, Vol. 30, No. 6, pp. 1109–1121. DOI: 10.1109/tkde.2017.2787640.
16. **Hu, L., Liu, H., Zhang, J., Liu, A. (2021).** KR-DBSCAN: A density-based clustering algorithm based on reverse nearest neighbor and influence space. Expert Systems with Applications, Vol. 186, pp. 115763. DOI: 10.10 16/j.eswa.2021.115763.
17. **Lai, W., Zhou, M., Hu, F., Bian, K., Song, Q. (2019).** A new DBSCAN parameters determination method based on improved MVO. IEEE Access, Vol. 7, pp. 104085–104095. DOI: 10.1109/access.20 19.2931334.
18. **Ozkok, F. O., Celik, M. (2017).** A new approach to determine EPS parameter of DBSCAN algorithm. International Journal of Intelligent Systems and Applications in Engineering, Vol. 4, No. 5, pp. 247–251. DOI: 10.18201/ijisae.2017533899.
19. **Gaonkar, M. N., Sawant, K. (2013).** AutoEpsDBSCAN: DBSCAN with EPS automatic for large dataset. International Journal on Advanced Computer Theoryand Engineering, Vol. 2, No. 2, pp. 2319–2526.

ISSN 2007-9737

Article received on 01/05/2023; accepted on 18/04/2024.
**Corresponding author is Nasereddine Amroune.*

ISSN 2007-9737

Optimizing the Performance of the IDS through Feature-Relevant Selection Using PSO and Random Forest Techniques

Benaissa Safa[1,*], Reda Mohamed-Hamou[2], Adil Toumouh[1]

[1] Djillali Liabes University, Computer Science Department, Algeria

[2] Dr. Tahar Moulay University, Computer Science Department, Algeria

benaissa.safa@univ-sba.dz, hamoureda@yahoo.fr, toumouh@gmail.com

Abstract. As the world becomes more digitalized, the potential for attacks increases, therefore, effective techniques for intrusion detection on network are needed. In this study, the authors propose a two steps approach. First, the Correlation-based Features Selection as a feature evaluator based on Particle Swarm Optimization is used to select the relevant features. This evaluator is compared with other evaluators. Second, the Random Forest algorithm is used to classify attacks in a network. A comparative study is also performed conducted with different classifiers such as Naïve Bayes, Stochastic Gradient Descent, Deep Learning, k-Nearest Neighbors and Support Vector Machine. Experiments were conducted on the NSL-KDD database and the results show an efficiency of 98.78% for binary classification. The performance results obtained show that the proposed technique performs better than other competing techniques.

Keywords. Classification, feature selection, intrusion detection system, machine learning, NSL-KDD data set, particle swarm optimization, random forest.

1 Introduction

Nowadays, the Internet has sparked a great technological revolution in terms of the exchange of information, knowledge and science between individuals and even institutions; at the same time, the use of the web has become one of the essential necessities of our daily life. Unfortunately, this dependence on the web has led some individuals to exploit it illegally through hacking, espionage, data theft, extortion and other malicious activities.

This reality poses a significant security threat to both individuals and companies. This issue is also becoming a real challenge for computer science researchers and developers.

Therefore, it is necessary to implement a security policy to protect company data and personal information from unexpected attacks. several tools are available to ensure data protection and personal information. The purpose of this protection is to reduce the risks associated with the confidentiality, integrity and availability of data.

An Intrusion Detection System (IDS) is considered to be the most important tool to ensure the functionality of computer security systems, because the IDS is the only tool that can guarantee the stability of the system, and then, because most attacks occur after an intrusion or by the injection of a malicious application. It is in charge of the response in the event of an attack as well as the stop or continuity strategies [8].

There are two main types of intrusion detection approaches in the literature: those based on scenarios (such as signature research, pattern matching, etc.) and those based on behavioral

Table 1. Summary table of some related works

Used algo/model	Data set	Classification	Accuracy (%)	Ref.
NDAE (DL - AE - RF)	NSL-KDD	5-class	85.42	[27]
	10% KddCup'99	5-class	97.85	
DNN - AE – SM	NSL-KDD	2-class, 5-class	-	[26]
DL - AE – SM (STL, SMR)	NSL-KDD	2-class, STL 2-class, SMR 5-class, STL 5-class, SMR	88.39 78.06 79.10 75.23	[19]
AE – DBN	10% KddCup'99	2-class	92.10	[3]
DBN	40% NSL-KDD	5-class	97.45	[15]
DBN	10% KddCup'99	5-class	93.49	[4]
DBN – LR	10% KddCup'99	5-class	97.90	[14]
DRBM	10% KddCup'99	2-class	94.00	
DNN	10% KddCup'99 NSL-KDD	2-class 5-class 2-class 5-class	93.00 93.50 80.10 78.50	[31]
RNN	NSL-KDD Test+ Test-21 Test+ Test-21	2-class 5-class	83.28 68.55 81.29 64.67	[35]
LSTM RNN	KddCup'99	5-class	97.54	[21]
HC + SVM	KddCup'99	5-class	95.72	[17]
CT + SVM	1998 DARPA	5-class	69.80	[20]
NB + KNN	NSL-KDD	5-class	84.86	[25]
KNN + SVM + PSO	KddCup'99	5-class	88.72	[1]
K-means + KNN	KddCup'99	5-class	99.01	[30]
GMMs + PSO + SVM	KddCup'99	5-class	99.99	[18]
FL + GA	10% KddCup'99	5-class	94.60	[10]
K-Means + NB + BNN	KddCup'99	5-class	99.90	[11]

approaches (for example, Bayesian analysis, statistical analysis and neural networks).

This last category aims to recognize abnormal behavior, compared to a definition or a modeling of normal or abnormal behaviors learned from a prior observation of the system, and in this case, learning seems possible. In contrast, in a scenario-based approach, the IDS relies on a pre-existing knowledge base referencing the various known attacks likely to be implemented in a computer system.

This knowledge is used by the IDS for the recognition of events produced by intrusion actions in the computer system that it observes. Therefore, this method requires regular updating of the knowledge base and the IDS focuses directly on the identification of misuse.

It is also possible to compare intrusion detection systems based on the data sources they rely on. Some IDS, known as HIDS (Host IDS) are based on the execution histories of specific programs or instruction sequences, which are often provided by the operating system but sometimes also by applications. Other IDS, typically known as NIDS (Network IDS), analyzes the packets sent over the network.

In theory, two response modes can be distinguished for IDS. Usually, a passive response

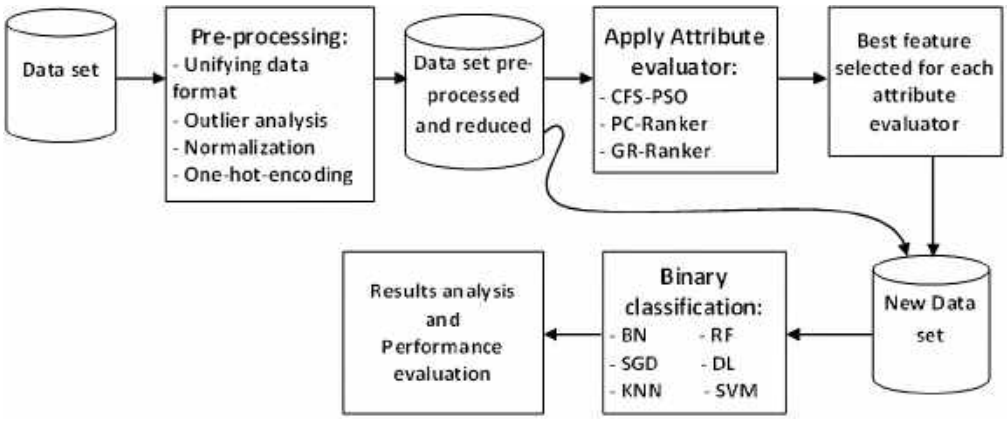


Fig. 1. Proposed research methodology

is adopted: the IDS broadcasts an alert and identifies the detected attack to an analysis or broadcast system by recording the detected intrusions in a log file.

However, more active responses should be considered, where the IDS aims to stop an attack at the moment of its detection by interrupting a connection or even counter-attacking [23].

In order to improve the efficiency of intrusion detection systems, several solutions have been proposed in this field. The authors remain focused on achieving this objective by conducting research on the use and integration of bio-inspiration techniques in general and particle swarm optimization (PSO) in particular.

PSO is a bioinspired optimization metaheuristic that was proposed by Eberhart and Kennedy in 1995 [12]. The technique of optimizing particle swarm was inspired by the collective behavior of birds or fish schools.

Each particle in the PSO is a fish or a bird in search space, with its own specific coordinates: position and velocity. Prior to searching for the optimum global position, particles try to maintain their local best positions [9]. In this paper, it is proposed to use the Correlation based Features Selection (CFS) feature evaluator, based on the bio-inspired technique of PSO, for selecting only the relevant features. Subsequently, the Random Forest (RF) classifier is chosen for attack classification in a network.

The RF algorithm is one of the most popular machine learning techniques. The sections of this article are arranged in the following order: Section 2 provides the related works in the field of intrusion detection systems, distinguishing, those that are based on machine learning methods and some others that focus on deep learning.

Section 3 presents the author's proposal, followed by a brief analysis of the KDDCup'99 data set and its versions used in this article, such as statistics and data preprocessing. This section concludes with a description of the different evaluation metrics used. Section 4 explores the analysis and discussion of the experimental results. Finally, section 5 presents a conclusion and future research directions suggested.

2 Related Work

Information security is an interesting area of research for its importance in the daily lives of individuals and even for institutions.

An intrusion detection system (IDS) is considered an important policy to improve the quality of computer security. In recent years, a considerable number of literature searches on intrusion detection have been published. In this section, a selection of this works is presented.
During the preceding decade, several studies have been done in the intrusion detection area, some of them based on machine learning methods and

Table 2. Composition of KDDCup'99 training data set (before and after preprocessing)

Connection Type	Before preprocessing	After preprocessing	
	No. of instances	No. of unique instances	Reduction (%)
DoS	3,883,370	247,267	93.63
Probe	41,102	13,860	66.28
R2L	1,126	999	11.28
U2R	52	52	00.00
T. Attacks	3,925,650	262,178	93.32
Normal	972,781	812,814	16.44
Total	4,898,431	1,074,992	78.05

Table 3. Composition of NSL-KDD data sets

Connection Type	Training set		Test set	
DoS	45,927	36.46%	7,458	33.08%
Probe	11,656	9.25%	2,421	10.74%
R2L	995	0.79%	2,754	12.22%
U2R	52	0.04%	200	0.89%
Total Attacks	58,630	46.54%	12,833	56.93%
Normal	67,343	53.46%	9,711	43.07%
Total	125,973	100%	22,544	100%

others focusing on deep learning. First, a few studies based on machine learning techniques are presented, followed by a few others based on deep learning.

2.1 IDS based on Machine Learning Techniques

In [6], the authors propose an algorithm for feature selection. The authors used these selected features to build an intrusion detection system based on the least squares support vector machine LSSVM-IDS.

They tested their experiment on three data sets such as KDDCup'99, NSL-KDD and Kyoto 2006+, and they showed that their algorithm gives improved accuracy per attack class. The paper presented by Altwaijry and Algarny in 2012 [5] explains the use of a Naïve Bayesian classifier for intrusion detection.

The authors evaluated their proposal by category of attacks on the 10 percent of KDDCup'99 and the corrected-KDD data set. In their article referenced by [22], the authors focused their work on the cluster center and nearest neighbor (CANN) approach to feature representation with the aim of detecting intrusions.

Table 4. Confusion matrix for binary classification

		Predicted class	
		Instance Normal	Attack
Actual class	Instance Normal	TN	FP
	Attack	FN	TP

They evaluated their experimentation on the KDDCup'99 data set. They used four types of attacks. In 2016, Han X. et al. [16] suggested principal component analysis for feature extraction and proposed an algorithm for intrusion detection based on the traditional Naïve Bayesian classification algorithm.

The authors used the 10 percent subset of KDDCup'99 (494,020 records, including 19.69 percent normal and 80.31 percent attack) to evaluate the performance of their solution.

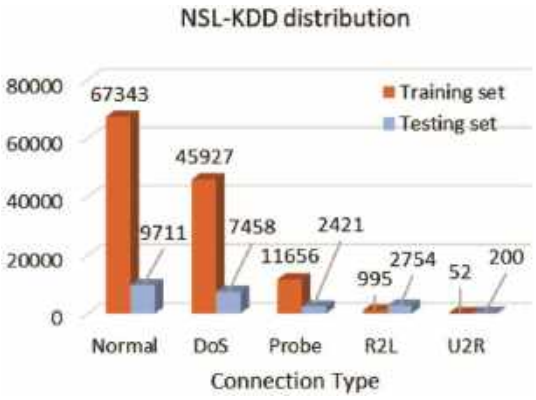
2.2 IDS based on Deep Learning Techniques

Since 2006, several studies on deep learning methods for intrusion detection have been published. the paper presented by Tang et al. in 2016 [28] explains an approach based on a deep neural network composed of an input layer of 6 dimensions, three hidden layers of 12, 6 and 3 neurons respectively and a 2-dimensional output layer.

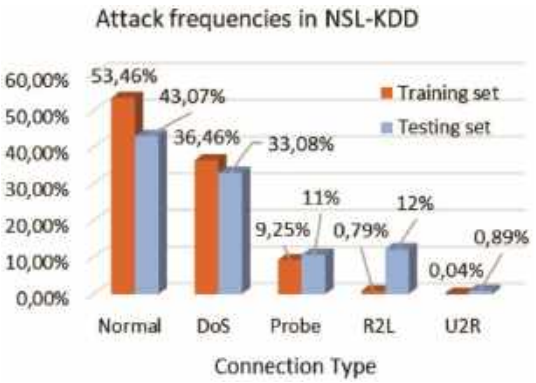
The authors tested their approach on the NSL-KDD data set and their model achieved an accuracy around 75.75 percent. The NDAE (Non-symmetric Deep Auto-Encoder) model, based on a Deep Auto-Encoder is proposed by Shone et al. in 2018 [27].

In this study, the number of attributes was reduced to 28 instead of a total of 41 attributes by this Auto-Encoder. The proposed model is composed of an input layer, six hidden layers and an output layer.

Their model is evaluated using the 10 percent subset of KDDCup'99 and NSL-KDD data set, and



(a) NSL-KDD distribution



(b) Attak frequencies in NSL-KDD

Fig. 2. Description of NSL-KDD data set

an accuracy of 97.85 percent and 85.42 percent for the two data sets respectively is obtained by the authors after using a random forest-based classifier for 5-class classification.

In 2016, Javaid et al. [19] developed a flexible and efficient NIDS (Network Intrusion Detection System) based on a proposed deep learning approach. The authors apply the technique of self-directed learning (STL).

They use the NSL-KDD data set to evaluate their system which achieved an accuracy rate of 88.39 percent and 79.10 percent for 2-class and 5-class respectively. In [4], the paper presented by authors explains the application of a Restricted Boltzmann Machine (RBM) and a Deep Belief Network (DBN) for a suggested deep learning approach to detect anomalies.

A feature reduction is performed by a first RBM. And the resulting weights are passed to a second RBM to create the DBN. the authors tested their approach on the KDDcup'99 data set, and their model showed improved accuracy (97.9 percent).

In 2017, Yin et al. [35] proposed an approach based on deep learning using a recurrent neural network (RNN-IDS). They chose the sigmoid function for activation and SoftMax as a classification function. The authors implemented their solution and tested it on the NSL-KDD data set.

The evaluation of their proposal shows an accuracy rate of 83.28 percent and 81.29 percent for a binary and multi-class (5-class) classification respectively on the KDDTest+ data set and an accuracy rate of 68.55 percent and 64.67 percent for a 2-class and 5-class respectively on the KDDTest-21 data set. A summary of some related works is shown in Table 1 below.

NDAE: Non-symmetric Deep Auto-Encoder; DL: Deep Learning; BNN: Back-propagation Neural Network; ANN: Artificial Neural Network; DNN: Deep Neural Network; RNN: Recurrent Neural Network; DBN: Deep Belief Network; DRBM: Discriminative Restricted Boltzmann Machine; AE: Auto-Encoder; SM: Soft-Max; SMR: Soft-Max Regression; STL: Self-Taught Learning; CT: Clustering Tree; LSTM: Long Short-Term Memory; GMMs: Gaussian Mixture Models; IDS: intrusion detection system; MDS: Malicious Detection System; NADS: Network Anomaly Detection System; LR: Logistic Regression; RF: Random Forest; HC: Hierarchical Clustering; NB: Naïve Bayes; K-Means; FL: Fuzzy Logic, GA: Genetic Algorithm; KNN: K-Nearest Neighbor; SVM: Support Vector Machine; PSO: Particle Swarm Optimization. SGD: Stochastic Gradient Descent.

3 Proposed Approach

As stated in some research, such as presented by Maniriho and Ahmad in 2018 [24], certain features have no influence in the attack detection process, or in other words, these unnecessary features may have a negative impact on attack determination performance.

Table 5. Feature set of KddCup'99 and NSL-KDD data set

No. f	Feature label	Type	No. f	Feature label	Type
Basic features class (B)			**Traffic 'same-Service' features class (TS)**		
1	Duration	Continuous	23	Count	Continuous
2	protocol_type	Symbolic	24	srv_count	Continuous
3	service	Symbolic	25	serror_rate	Continuous
4	flag	Symbolic	26	srv_rerror_rate	Continuous
5	src_bytes	Continuous	27	serror_rate	Continuous
6	dst_bytes	Continuous	28	srv_rerror_rate	Continuous
7	land	Symbolic	29	same_rerror_rate	Continuous
8	wrong_fragment	Continuous	30	diff_srv_rate	Continuous
	urgent	Continuous	31	srv_diff_host_rate	Continuous
Content features class (C)			**Traffic 'same-Host' features class (TH)**		
10	Hot	Continuous	32	dst_host_count	Continuous
11	num_failed_logins	Continuous	33	dst_host_srv_count	Continuous
12	logged_in	Symbolic	34	dst_host_same_srv_rate	Continuous
13	num_compromised	Continuous	35	dst_host_diff_srv_rate	Continuous
14	root_shell	Continuous	36	dst_host_same_src_port_rate	Continuous
15	su_attempted	Continuous	37	dst_host_srv_diff_host_rate	Continuous
16	num_root	Continuous	38	dst_host_serror_rate	Continuous
17	num_file_creations	Continuous	39	dst_host_srv_serror_rate	Continuous
18	num_shells	Continuous	40	dst_host_rerror_rate	Continuous
19	num_access_files	Continuous	41	dst_host_srv_rerror_rate	Continuous
20	num_outbound_cmds	Continuous			
21	is_host_login	Symbolic			
22	is_guest_login	Symbolic			

The study described in this section aims to propose an IDS model based on the machine learning methods for the attack detection, based on the features selection that has an important influence in the attack determination process.

To achieve this objective and select only the relevant features for training of the proposed model, various feature evaluators were employed by conducting multiple tests.

Three evaluators, namely Correlation based Features Selection (CFS), Pearson's Correlation (PC) and Gain Ratio (GR), were the focus of these test.

The ranking scores generated by feature class using these three evaluators were used to select twenty-one considered as relevant out of a total of forty-one. The proposed model is illustrated in the block diagram in Figure 1 below.

After is a brief description of the three evaluators used.

3.1 Correlation based Feature Selection

The principle of the Correlation based Features Selection (CFS) is to measure Pearson's correlation between an attribute and the class, it

Table 6. Features selected by different techniques for binary classification

Attribute Evaluator: **Search Method:**	**CFS** **PSO**	**Pearson's Correlation** **Ranker**	**Gain Ratio** **Ranker**
Features class	Position of the 21 Best selected features	Position of the 21 Best selected features	Position of the 21 Best selected features
Basic (B)	1, 3, 4, 5, 6, 7	3, 4, 8	3, 4, 5, 6, 8
Content (C)	12, 14, 15, 16, 21, 22	12	12
Traffic 'same-Service' (TS)	26, 27, 29, 30	23, 25, 26, 27, 28, 29, 30, 31	23, 25, 26, 28, 29, 30, 31
Traffic 'same-Host' (TH)	34, 35, 37, 38, 39	32, 33, 34, 35, 36, 38, 39, 40, 41	32, 33, 34, 35, 37, 38, 39, 41

determines the value of the attribute. By treating each value as an indicator, nominal properties are evaluated individually.

A weighted average is used to determine the overall correlation of a nominal attribute. The particle swarm optimization method is chosen as the search method for this feature evaluator. This approach was invented by Eberhart and Kennedy in 1995 [12]. The principle of PSO is population-based, which aims to find a sub-optimal solution in the search space.

At each iteration of the PSO algorithm, each individual (particle X_i) changes and updates by the two best values, the best solution (local position) based on its speed that the particle X_i has obtained so far and the best position global [13].

3.2 Pearson's Correlation

The Pearson coefficient indicator denoted r is a measure used to detect the presence or absence of a linear relationship between two variables.

The value of this measure of correlation varies from -1 to $+1$, a positive measure indicates that the two variables vary together in the same direction, when the value of r is close to $+1$, we say that there is a strong correlation.

While a negative measure indicates that one variable increases, the other decreases and when this value is close to -1, we say that there is a strong negative correlation, we say that an absence of a relationship between two variables if r takes 0 value. We remind that the formula (1) below to calculate the Pearson correlation coefficient:

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}. \quad (1)$$

3.3 Gain Ratio

An extension of information gain, called gain ratio, was used to select the best feature feature for splitting the dataset. The gain ratio is calculated by normalizing the information gain with aid of division information.

A feature will be favored by the gain of information if it has a large number of values. The Gain Ratio (GR) is calculated as follows:

$$\mathrm{GR}(S, f_j) = \frac{\mathrm{IG}(S, f_j)}{\mathrm{SI}(S, f_j))}, \quad (2)$$

where:
IG: is the Information Gain.
SI: is the Split Information can be calculated as follows:

$$\mathrm{SI}(S, f_j) = -\sum_{S_{jk} \in S_j} (\frac{|S_{jk}|}{|S|} * log_2(\frac{|S_{jk}|}{|S|})), \quad (3)$$

where:
C: a set of classes.
S: a training data set.
$\overrightarrow{f}$: a feature vector.
S_j : a hyper-set containing sets with the same values of the feature f_j.
$IG(S, f_j)$: the information gain by splitting the dataset S with the feature f_j.

Table 7. DR and FAR of different classifiers when using different feature selection evaluators

	DR			**FAR**		
ML	CFS-PSO	PC-Ranker	GR-Ranker	CFS-PSO	PC-Ranker	GR-Ranker
NB	0.6977	0.6892	0.6889	0.0485	0.0461	0.0483
RF	0.9884	0.9851	0.9887	0.0130	0.0288	0.0158
SGD	0.9056	0.9605	0.9495	0.0460	0.0745	0.0764
DL	0.9107	0.9518	0.9522	0.0491	0.0716	0.0730
KNN	0.9801	0.9813	0.9792	0.0268	0.0355	0.0278
SVM	0.9108	0.9606	0.9476	0.0721	0.0738	0.0754

3.4 Dataset Description

Research in the field of intrusion detection (ID) requires the use of data sets to evaluate the efficiency and effectiveness of the proposed solutions by researchers in order to achieve concrete objectives. In this context, there are a variety of freely accessible network-based data sets available for intrusion detection research.

Among these data sets, we focused on the KDDCup'99 data set in our work; this data set is primarily concerned with intrusion detection and was constructed and modified from original network traffic data collected by the DARPA 1998 evaluation program under the supervision of the Massachusetts Institute of Technology (MIT) Lincoln Laboratory.

The data set in question is often used in the literature and comported of around 4,900,000 connection records, each of which is composed of 41 values and is labeled as either normal or an attack, each value corresponding to a different feature [29]. The KDDCup'99 data set can be listed as a normal traffic class and four categories to group the different kinds of attacks as shown below:

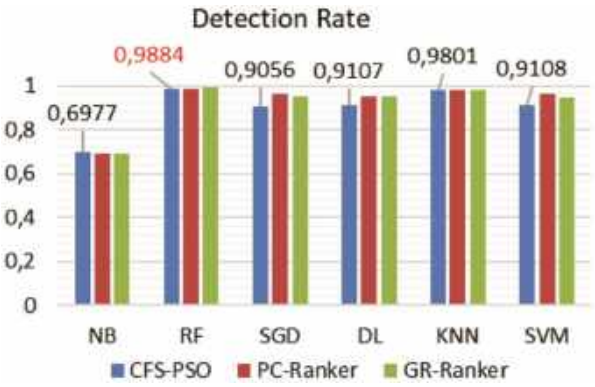
- Normal: it indicates that the network traffic record is normal or benign.
- Denial of Service attack (DoS): an intrusion or a kind of attack that tries to make some computing resources (server, host, memory, ...) inaccessible for the client, such as memory that is too full, with the objective of using the victim's resources.
- Probing attack (Probe): this category of attack includes all kinds of malicious activity, in which the perpetrator gathers detailed information about the system infrastructure and its security configurations, and for the goal by passing the firewall and conducting critical attacks.
- Remote to Local attack (R2L): the intruder does not belong to the computer network, but sends packets to the server or to another machine as a local user in order to gain access.
- User to Root attack (U2R): after several attempts to access network resources, the intruder has the character of a legitimate or normal user. Then, it attempts to access root or superuser privileges.

In 2009, Tavallaee et al. [29] provided and developed a new refined and improved version of the KDDCup'99 corpus under the appellation NSL-KDD.

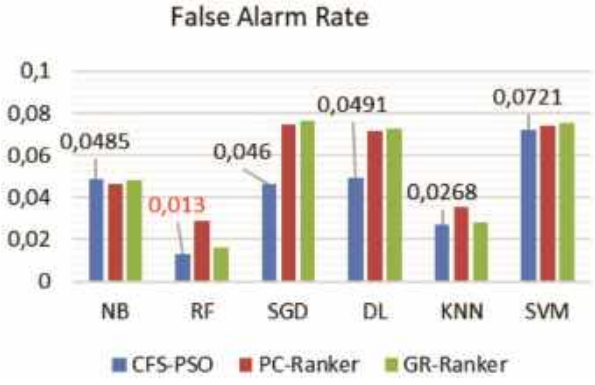
For security researchers, the number of publicly available data sets for network IDS (NIDS) is limited. KDDCup'99 and NSL-KDD are the most widely utilized and publicly available data sets for testing the effectiveness of different existing and newly announced machine learning methods [32].

In this paper, the NSL-KDD data set is used to train and test the proposed solution for intrusion detection. This version of the data set is derived from the main KDDCup'99 data set.

It reduced and improved the data set version which contains 125,973 instances. A brief description of these data sets is reported in Table



(a) Detection rate



(b) False Alarm rate

Fig. 3. Performance evaluation of different classifiers when using different feature selection evaluators

2 and Table 3. The different connections types for KddCup'99 and NSL-KDD data set are:

- Probe (Probing): ipsweep, nmap, portsweep, satan.
- DoS (Denial of Service): back, land, neptune, pod, smurf, teardrop.
- U2R (User to Root): buffer_overflow, loadmodule, perl, rootkit.
- R2L (Remote to Local): ftp_write, guesspasswd, imap, multihop, phf, spy, warezclient, warezmaster.

3.5 Data Preprocessing

As mentioned above, the KDDCup'99 and NSL-KDD data sets gather 41 features of different types and are distributed as follows, three of a nominal type such as 'Protocol type', 'Service' and 'Flag', four are binary and the thirty-four remaining features are of continuous type.

Knowing that most of the algorithms and methods only work with numbers and in order to obtain better results from experiments, a preprocessing must be performed on the data sets.

Firstly, using the One-hot-encoding [36] for transformed the nominal features to discrete features, for example, dummy variables are used to encode the textual values of the 'Protocol Type' feature (i.e. [1,0,0], [0,1,0], [0,0,1] for tcp, udp, icmp), knowing that the nominal features 'Protocol type', 'Service' and 'Flag' of the 10% KDDCup'99 training data set have 3, 66 and 11 categories respectively.

Secondly, another main step to complete is the standard normalization, also called standardization or z-score normalization. The purpose of this step is to scale all features in order to guarantee that all predictor values are on the same scale.

The principle of z-score normalization is to subtract from the data their empirical mean μ and divide them by their standard deviation σ. In this case, we apply the formula of equation (1) shown below.

Such that, for each feature j, $\mu(j)$ and $\sigma(j)$ denote respectively the mean and the standard deviation of the data vector X^j of the feature j, where each value of the vector X_i^j is transformed according to equation (4):

$$X_i^j = \frac{X_i^j - \mu(j)}{\sigma(j)}. \tag{4}$$

During the data set preprocessing phase, the training and testing databases in the KDDCup'99 collection have a multitude of duplicate instances.

This duplication represents one of the main disadvantages of this data set. These redundancies have a negative impact on the results of the experiments, and must therefore be removed. It is noted that, the training and test data sets, respectively, had about 78.05 percent and 80.68 percent of duplicated instances [29], (see Table 2).

Table 8. Precision and accuracy rate of different classifiers when using different feature selection evaluators

	Precision			**Accuracy**		
ML	CFS-PSO	PC-Ranker	GR-Ranker	CFS-PSO	PC-Ranker	GR-Ranker
NB	0.9500	0.9518	0.9496	0.8070	0.8032	0.8021
RF	0.9902	0.9783	0.9881	0.9878	0.9791	0.9868
SGD	0.9630	0.9446	0.9426	0.9264	0.9454	0.9383
DL	0.9608	0.9462	0.9452	0.9280	0.9417	0.9413
KNN	0.9797	0.9733	0.9790	0.9771	0.9741	0.9762
SVM	0.9435	0.9450	0.9432	0.9182	0.9458	0.9377

Often, in the preprocessing procedure for data sets, it is also important to remove records that contain incorrect values in the fields, such as character strings arranged in numerical fields or vice versa, missing values, etc.

After preprocessing the KDDCup'99 databases. It was noticed that 4,898,431 records which constitute the initial training set was reduced to 1,074,992 unique data points due to redundancy, this significant reduction represents a rate of 78.05 percent as shown in Table 2.

Similarly, for the KDDCup99's test set, it was noted that, a total number of 2,984,154 data points was reduced to 576,449 unique instances which represents a reduction rate of 80.68 percent. The results of this table (Table 3) are interpreted in Figures 2a and 2b below.

As previously stated, it is noted that all instances of the same KDDCup'99 data set or its derivatives are composed of 41 features. each feature has only one type of continuous, discrete or symbolic variable [33].

Generally, features are divided into four aspects or classes (see Table 5), the first nine features relate to basic intrinsic properties of the network connection, such as connection duration, protocol type, network service (http, telnet, etc.), etc.

Are grouped to form a first aspect or base class (B). The following thirteen features correspond to domain knowledge or the content of a network connection. The purpose of the content aspect features (C) is to assess the payload of the original TCP packets and to detect attacks that are hidden and not commonly present such as those of the U2R and R2L classes.

In this case, to identify such attacks, the researchers retrieved information on the amount of login failures, which suggest intrusive behavior [34]. The other two classes are encapsulated under the name Traffic; this large traffic aspect groups features which are called time-based and calculated with respect to a time interval.

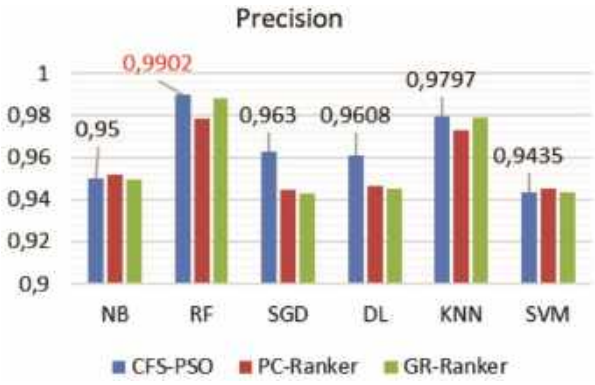
The first of the traffic aspects includes the "same service" (TS) features, consists to examine only connections established during the last two continuous seconds which have the same service as the present connection.

The second traffic aspect includes the last ten features that are from "same host" (TH), consists of an analysis of the connections made in the last continuously two seconds which have the identical final host as the present connection in order to calculate the behavioral statistical properties of the network connection, relating to protocol, serving, etc. [2].

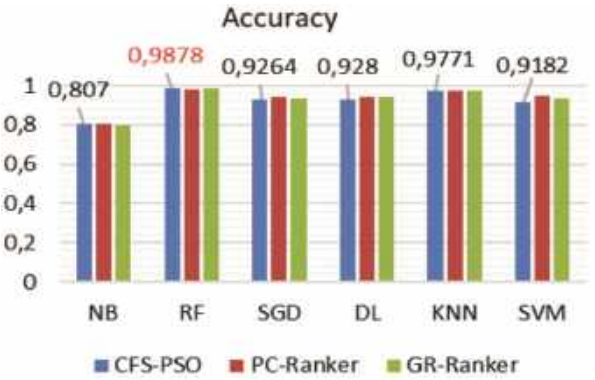
3.6 Evaluation Criteria

Generally, to evaluate the IDS detection precision, the following measures are often used:

– True Positive (TP): this metric represents the number of attacks detected and correctly classified by the model.
– True Negative (TN): a metric that indicates the number of normal instances predicted and correctly classified as normal traffic.



(a) Precision



(b) Accuracy

Fig. 4. Precision and accuracy rate of different classifiers when using different feature selection evaluators

– False Positive (FP): this metric represents the number of normal instances recognized and incorrectly classified as attacks by the model.
– False Negative (FN): a metric that indicates the number of attacks predicted and incorrectly classified as normal traffic by the model.

These metrics often form the confusion matrix values shown in Table 4 below for a binary classification problem.

Other measures were used that can be calculated based on the values of this confusion matrix as presented in Table 4, as follows: Detection Rate (DR) or True Positive Rate (TPR):

$$\mathrm{DR} = \frac{TP}{TP + FN}. \tag{5}$$

False Alarm Rate (FAR) or False Positive Rate (FPR):

$$\mathrm{FAR} = \frac{FP}{TN + FP}. \tag{6}$$

Precision:

$$\mathrm{Precision} = \frac{TP}{TP+FP}. \tag{7}$$

Overall accuracy is defined as the proportion of instances in a set of occurrences that have been correctly classified. This metric is less useful in the case where there is a significant imbalance between the classes:

$$\mathrm{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{8}$$

4 Experiment Results and Discussion

After applying the three attribute evaluation metrics (CFS-PSO, PC-Ranker, GR-Ranker), the results obtained for binary classification are shown in Table 6. Therefore, for each feature class, it is also important to choose the most relevant or influential features for the intrusion detection process.

So, the most relevant features are chosen for each class (Basic: B, Continent: C, Traffic same Service: TS and Traffic same Host: TH) by following the order of features based on their order of merit in their respective classes. These features are presented in Table 6.

For example, if CFS-PSO technique used in binary classification case, the best features selected for Basic Class are (Duration, service, flag, src_bytes, dst_bytes, land).

4.1 Analysis of Experimental Results

After selecting the top twenty-one features for each attribute evaluator from the entire data set. In the binary classification experiments, the resulting data set can be trained and tested using a variety of machine learning techniques, such as Naïve Bayes, Random Forest, Stochastic Gradient Descent, Deep Learning, K-Nearest Neighbors and Support Vector Machine.

The obtained results are presented below. Based on the corresponding new NSL-KDD data

Table 9. Comparison of the results with other algorithms (NSL-KDD data set used)

Method	Accuracy (%)	Ref.
DL-AE-SM(SMR)	78.06	[19]
DL-AE-SM(STL)	88.39	
DNN	80.1	[31]
RNN	83.28	[35]
MI-BGSA	88.36	[7]
Proposed-method	98.78	

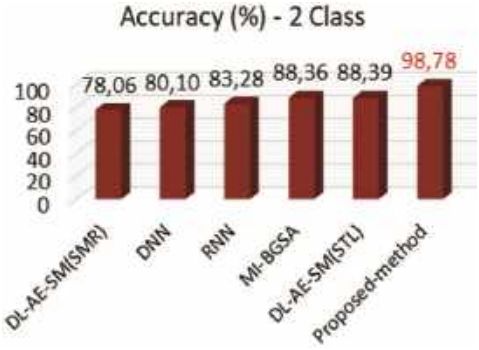


Fig. 5. Comparison of accuracy rate with other algorithms (2-class) which NSL-KDD used

set, which contains only the twenty-one best selected features for each attribute evaluator (presented in Table 6), various performance measures can be calculated such as DR, FAR, precision and system accuracy, based on the results of the confusion matrix.

Table 7 presents the obtained results of the Detection Rate and False Alarm Rate measurements of each of the machine learning techniques and for each used attribute evaluator. These results are interpreted in Figures 3a and 3b.

In the same way, precision and accuracy measurements can be calculated. Table 8 presents the obtained results of precision and accuracy measurements of each machine learning techniques and for each used attribute evaluator. The results of this table are interpreted in Figures 4a and 4b.

Finally, Table 9 shows a performance comparison of the proposed method with some other recent methods using the same data set (NSL-KDDTest) in terms of accuracy. It can be seen from the table that the proposed method (CFS-PSO + RF) ranks first in terms of accuracy in binary classification case.

Therefore, the proposed CFS-PSO attribute evaluator-based RF classifier performs better than all other competitive techniques for binary classification case (see Figure 5).

4.2 Discussion of Experimental Results

In a data set, applying a method for eliminating unnecessary features is indispensable because these extra features decrease the precision and efficiency of the prediction algorithms. Additionally, as the number of features in a data set grows, so does the searchable space.

In this research, feature selection and reduction were performed by keeping only the most relevant features. To accomplish this, three attribute evaluation metrics were applied: CFS-PSO, PC-Ranker and GR-Ranker in binary classification. The results are shown in Table 6.

In order to improve the DR and optimize the performance of the IDS, the three attribute evaluation metrics can be applied to the data set, by selecting the same number of relevant features for each of these metrics.

After running several tests, twenty-one relevant features were selected. Various performance measures were calculated, including DR, FAR, precision and system accuracy, based on the results of the confusion matrix, the obtained results are discussed as follows: In the binary classification case, the performance comparison results are shown in Tables 7 and 8, which indicate that the proposed technique (CFS-PSO attribute evaluation metric combined with RF classifier) achieved a higher DR of 98.84%, while the False Alarm Rate (FAR is 1.3%) is also the lowest compared to other machine learning techniques. In terms of precision and accuracy, Figures 4a and 4b also show a comparison of performances and prove that the proposed method takes the first place with a precision rate of 99.02% and an accuracy rate of 98.78%.

5 Conclusion and Future Work

This paper discusses an effective intrusion detection technique that is divided into two phases. In the first phase, relevant features were selected by eliminating those that do not have a significant influence on the intrusion detection procedure.

This was achieved by using an attribute elevator technique called Correlation based Features Selection (CFS) technique based on the Particle Swarm Optimization (PSO) method, resulting in a feature space reduction of approximately 50%.

In the second phase, the proposed classification algorithm Random Forest (RF) and different machine learning algorithms were tested to evaluate the performance of the proposed method, experiments were conducted on the new NSL-KDD data set containing only twenty-one features.

The experiments carried in this study are divided into three classes, Firstly, a comparison is made between the chosen attribute evaluator (CFS-PSO) and two other evaluators, such as PC-Ranker and GR-Ranker, in the second set of experiment, a comparison is made between the proposed classifier (RF) and other machine learning classifiers, namely NB, SGD, DL, KNN and SVM.

The experimental results on the NSL-KDD data set show the promising performance of the proposed techniques in terms of accuracy and detection rate compared to competitive methods.

In the final class of experiments, the proposed technique is compared to different previously existing methods.

The obtained performance results indicate that the proposed technique outperforms other methods in the binary classification. Finally, it should be noted that the current study has two major limitations, namely real-time operation and the ability to detect zero-day attacks.

To address these limitations and further improve the proposed technique, future work could focus on finding more efficient solutions for detecting zero-day attacks and developing an IDS that works in real-time. It is recommended to test the technique on other data sets such as UNSW-NB15, CSE-CIC-IDS2018.

References

1. **Aburomman, A. A., Reaz, M. B. I. (2016).** A novel SVM-kNN-PSO ensemble method for intrusion detection system. Applied Soft Computing, Vol. 38, pp. 360–372. DOI: 10.1016/j.asoc.2015.10.011.
2. **Aggarwal, P., Sharma, S. K. (2015).** Analysis of KDD dataset attributes - class wise for intrusion detection. Procedia Computer Science, Vol. 57, pp. 842–851. DOI: 10.1016/j.procs.2015.07.490.
3. **Alom, M. Z., Bontupalli, V., Taha, T. M. (2016).** Intrusion detection using deep belief networks. 2015 National Aerospace and Electronics Conference (NAECON), pp. 339–344. DOI: 10.1109/NAECON.2015.7443094.
4. **Alrawashdeh, K., Purdy, C. (2016).** Toward an online anomaly intrusion detection system based on deep learning. 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 195–200. DOI: 10.1109/ICMLA.2016.0040.
5. **Altwaijry, H., Algarny, S. (2011).** Bayesian based intrusion detection system. IAENG Transactions on Engineering Technologies: Special Edition of the World Congress on Engineering and Computer Science, Vol. 1, pp. 29–44. DOI: 10.1016/j.jksuci.2011.10.001.
6. **Ambusaidi, M. A., He, X., Nanda, P., Tan, Z. (2016).** Building an intrusion detection system using a filter-based feature selection algorithm. IEEE Transactions on Computers, Vol. 65, No. 10, pp. 2986–2998. DOI: 10.1109/TC.2016.2519914.
7. **Bostani, H., Sheikhan, M. (2017).** Hybrid of binary gravitational search algorithm and mutual information for feature selection in intrusion detection systems. Soft Computing, Vol. 21, No. 9, pp. 2307–2324. DOI: 10.1007/s00500-015-1942-8.
8. **Boudia, A., Hamou, R. M., Amine, A. (2017).** A new meta-heuristics for intrusion detection system inspired from

the protection system of social bees. International Journal of Information Security and Privacy, Vol. 11, No. 1, pp. 18–34. DOI: 10.4018/IJISP.2017010102.

9. **Bousmaha, R., Hamou, R. M., Amine, A. (2022).** Optimizing connection weights in neural networks using hybrid metaheuristics algorithms. International Journal of Information Retrieval Research, Vol. 12, No. 1, pp. 1–21. DOI: 10.4018/ijirr.289569.
10. **Chadha, K., Jain, S. (2015).** Hybrid genetic fuzzy rule based inference engine to detect intrusion in networks. Intelligent Systems and Computing, Vol. 321, pp. 185–198. DOI: 10.1007/978-3-319-11227-5_17.
11. **Dubey, S., Dubey, J. (2015).** KBB: A hybrid method for intrusion detection. 2015 International Conference on Computer, Communication and Control (IC4), pp. 1–6. DOI: 10.1109/IC4.2015.7375704.
12. **Eberhart, R., Kennedy, J. (1995).** A new optimizer using particle swarm theory. MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43. DOI: 10.1109/MHS.1995.494215.
13. **Elngar, A., Mohamed, D. A., Ghaleb, F. F. (2013).** A real-time anomaly network intrusion detection system with high accuracy. Information Sciences Letters, Vol. 2, No. 2, pp. 49–56. DOI: 10.12785/isl/020201.
14. **Fiore, U., Palmieri, F., Castiglione, A., De-Santis, A. (2013).** Network anomaly detection with the restricted boltzmann machine. Neurocomputing, Vol. 122, pp. 13–23. DOI: 10.1016/j.neucom.2012.11.050.
15. **Gao, N., Gao, L., Gao, Q., Wang, H. (2014).** An intrusion detection model based on deep belief networks. 2014 Second International Conference on Advanced Cloud and Big Data, pp. 247–252. DOI: 10.1109/CBD.2014.41.
16. **Han, X., Xu, L., Ren, M., Gu, W. (2015).** A naive bayesian network intrusion detection algorithm based on principal component analysis. 2015 7th International Conference on Information Technology in Medicine and Education (ITME), pp. 325–328. DOI: 10.1109/ITME.2015.29.
17. **Horng, S. J., Su, M. Y., Chen, Y. H., Kao, T. W., Chen, R. J., Lai, J. L., Perkasa, C. D. (2011).** A novel intrusion detection system based on hierarchical clustering and support vector machines. Expert Systems with Applications, Vol. 38, No. 1, pp. 306–313. DOI: 10.1016/j.eswa.2010.06.066.
18. **Hu, W., Gao, J., Wang, Y., Wu, O., Maybank, S. (2014).** Online adaboost-based parameterized methods for dynamic distributed network intrusion detection. IEEE Transactions on Cybernetics, Vol. 44, No. 1, pp. 66–82. DOI: 10.1109/TCYB.2013.2247592.
19. **Javaid, A., Niyaz, Q., Sun, W., Alam, M. (2016).** A deep learning approach for network intrusion detection system. Proceedings of the 9th EAI International Conference on Bio-Inspired Information and Communications Technologies (Formerly BIONETICS), pp. 21–26. DOI: 10.4108/eai.3-12-2015.2262516.
20. **Khan, L., Awad, M., Thuraisingham, B. (2007).** A new intrusion detection system using support vector machines and hierarchical clustering. The VLDB Journal, Vol. 16, pp. 507–521. DOI: 10.1007/s00778-006-0002-5.
21. **Kim, J., Kim, H. (2017).** An effective intrusion detection classifier using long short-term memory with gradient descent optimization. 2017 International Conference on Platform Technology and Service (PlatCon), pp. 1–6. DOI: 10.1109/PlatCon.2017.7883684.
22. **Lin, W. C., Ke, S. W., Tsai, C. F. (2015).** CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. Knowledge-Based Systems, Vol. 78, pp. 13–21. DOI: 10.1016/j.knosys.2015.01.009.

23. **Lokbani, A. C., Lehireche, A., Hamou, R. M., Boudia, M. A. (2015).** An approach based on social bees for an intrusion detection system by scenario. Securing the Internet of Things: Concepts, Methodologies, Tools, and Applications, pp. 914–938. DOI: 10.4018/978-1-5225-9866-4.ch040.

24. **Maniriho, P., Ahmad, T. (2018).** Analyzing the performance of machine learning algorithms in anomaly network intrusion detection systems. 2018 4th International Conference on Science and Technology (ICST), pp. 1–6. DOI: 10.1109/ICSTC.2018.8528645.

25. **Pajouh, H. H., Javidan, R., Khayami, R., Dehghantanha, A., Choo, K. K. R. (2019).** A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. IEEE Transactions on Emerging Topics in Computing, Vol. 7, No. 2, pp. 314–323. DOI: 10.1109/TETC.2016.2633228.

26. **Potluri, S., Diedrich, C. (2016).** Accelerated deep neural networks for enhanced intrusion detection system. 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), Vol. 2016-November, pp. 1–8. DOI: 10.1109/ETFA.2016.7733515.

27. **Shone, N., Ngoc, T. N., Phai, V. D., Shi, Q. (2018).** A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, Vol. 2, No. 1, pp. 41–50. DOI: 10.1109/TETCI.2017.2772792.

28. **Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., Ghogho, M. (2016).** Deep learning approach for network intrusion detection in software defined networking. 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), pp. 258–263. DOI: 10.1109/WINCOM.2016.7777224.

29. **Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A. (2009).** A detailed analysis of the KDD CUP 99 data set. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pp. 1–6. DOI: 10.1109/CISDA.2009.5356528.

30. **Tsai, C. F., Lin, C. Y. (2010).** A triangle area based nearest neighbors approach to intrusion detection. Pattern Recognition, Vol. 43, No. 1, pp. 222–229. DOI: 10.1016/j.patcog.2009.05.017.

31. **Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., Venkatraman, S. (2019).** Deep learning approach for intelligent intrusion detection system. IEEE Access, Vol. 7, pp. 41525–41550. DOI: 10.1109/ACCESS.2019.2895334.

32. **Vinayakumar, R., Soman, K. P., Poornachandran, P. (2017).** Evaluating effectiveness of shallow and deep networks to intrusion detection system. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1282–1289. DOI: 10.1109/ICACCI.2017.8126018.

33. **Wang, G., Hao, J., Mab, J., Huang, L. (2010).** A new approach to intrusion detection using artificial neural networks and fuzzy clustering. Expert Systems with Applications, Vol. 37, No. 9, pp. 6225–6232. DOI: 10.1016/j.eswa.2010.02.102.

34. **Xiao, Y., Xing, C., Zhang, T., Zhao, Z. (2019).** An intrusion detection model based on feature reduction and convolutional neural networks. IEEE Access, Vol. 7, pp. 42210–42219. DOI: 10.1109/ACCESS.2019.2904620.

35. **Yin, C., Zhu, Y., Fei, J., He, X. (2017).** A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, Vol. 5, pp. 21954–21961. DOI: 10.1109/ACCESS.2017.2762418.

36. **Zhang, Q., Bao, H., You, Y., Lee, K., Guo, D. (2018).** Category coding with neural network application. arXiv. DOI: 10.48550/arXiv.1805.07927.

ISSN 2007-9737

Article received on 18/04/2023; accepted on 23/04/2024.
** Corresponding author is Benaissa Safa.*

ISSN 2007-9737

Mining a Trending Topic: U.S. Immigration on the Context of Social Media

Esteban Castillo[1,*], Ofelia Cervantes[2]

[1] Tecnológico de Monterrey,
Escuela de Ingeniería y Ciencias,
Mexico

[2] Universidad de las Américas Puebla,
Department of Computer Science,
Mexico

esteban.castillojz@tec.mx, ofelia.cervantes@udlap.mx

Abstract. This paper presents a text mining approach for extracting valuable patterns from social media documents in the context of U.S. immigration. The paper points out the uncovering of statistical features alongside linguistic elements based on graph techniques. The use of graphs provide rich data structures for representing lexical and syntactic aspects of texts, allowing the discovery of complex patterns that used by experts could provide valuable insight. The proposed method is applied over a Twitter-X/-Reddit dataset that comprise English and Spanish language samples from 2016 up to 2019. Experimental results showed that our interpretation of classic statistic techniques provide a baseline understanding of the topic while a more robust analysis (graphs) permits to uncover/predict hidden patterns over large amount of samples. In particular, the use of a co-occurrence graph helped to obtain relevant words, phrases and sentences while a user-interaction graph allow to detect important users, communities and interactions among themselves.

Keywords. Text mining, statistics, graph mining, social network analysis, natural language processing, big data.

1 Introduction

Social media sites are an essential information resource related to every topic/domain around the world. Part of their success is due to the inherent openness for public consumption, clean and structured data, rich developer tooling, and broad appeal to users from every walk of life. Among the vast amount of available data on this sites, finding what is trending and how it is being discussed has emerged as an essential tool for understanding how people connect and how they share ideas, attitudes and even media consumption toward specific topics.

Text mining methods arrived as an optimal solution for acquiring, analyzing and predicting textual patterns from large amounts of data on social media. Mining methods, provide insightful knowledge that can be used by different domain experts for understanding user-profiles, authorship-styles, demographic information, sentiment polarity and even complex patterns related to the semantics of data.

Different studies around text mining have showed the impact of statistical and social network analysis for detecting insightful knowledge. Despite the progress achieved, there are still opportunities to create alternative approaches for representing and extracting complex patterns based on the combination of classic statistics and graph representations.

Considering the above, this paper proposes a text mining approach for identifying text patterns based on statistics and graph mining techniques over a highly commented topic (U.S. immigration).

ISSN 2007-9737

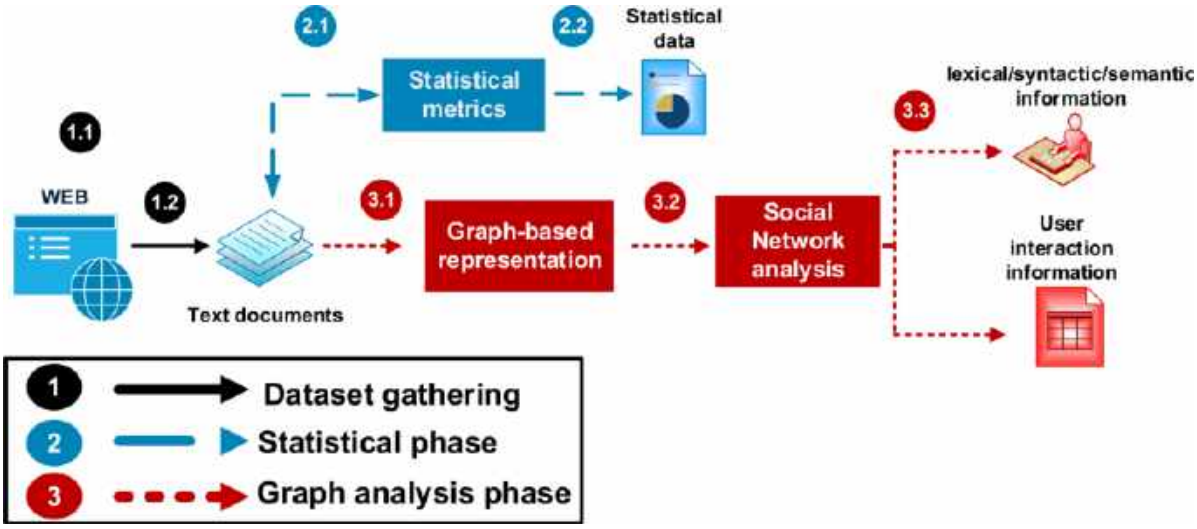


Fig. 1. Proposed text mining process

The **approach contribution** relies on the extraction/understanding of patterns and the creation of graph-based representations to detect knowledge using social network analysis tools. The **hypothesis** is that statistic techniques combined with more refined data structures (graphs) could be suitable for detecting representative elements over texts.

The **approach motivation** is to provide a valuable guide for mining a topic by using distinct text tools that normally are not combine together over a specific problem. The remainder of this paper is structured as follows:

Section 2 present existing approaches that deal with the extraction of knowledge from social media by using text mining techniques. Sections 3 to 6 provide details and examples on the design and implementation of the approach. Finally, implications and conclusions derived from this work thus far are presented in Section 7.

2 Related Work

Many literature deals with the extraction and digesting of social media on different topics and domains [3]. Most of it, is based on the use of classic statistic metrics, Information Retrieval (IR), Natural Language Processing (NLP), Machine learning (ML) and ultimately text mining methods. This literature, is highly dense and cover different applications and methodology approaches. Therefore, related work could be seen from two main perspectives: overall text mining approaches and specific research avenues related to social media on the context of political documents.

2.1 Text Mining

Text mining is the process of transforming unstructured text into a structured format to identify meaningful patterns and new insights in data [21]. The process of text mining involves the use of different research methods for obtaining valuable information from large amounts of data [27].

Some of this methods include techniques for acquiring and analyzing digital documents with NLP or statistics (besides others) [31, 5]. Researchers today, are using distinct text mining approaches for predicting domain patterns, public opinion and collectible behavior [13].

Text mining software has impacted the way that many industries work, allowing them to improve user experience and business decisions. Examples of this impact can be seen in different areas like customer service [23] where chatbots, and profiling tools are making the user experience

Table 1. Immigration keywords ordered by frequency of occurrence

English keywords obtained			
immigration$_1$	migration$_2$	naturalization$_3$	deportation$_4$
passport$_5$	greencard$_6$	border$_7$	trump$_8$
embassy$_9$	patrol$_{10}$	mexican$_{11}$	american$_{12}$
biden$_{13}$	workforce$_{14}$	alien$_{15}$	
Spanish keywords obtained			
inmigracion$_1$	migrante$_2$	indocumentado$_3$	repatriacion$_4$
deportacion$_5$	paisano$_6$	remesa$_7$	pasaporte$_8$
mexico$_9$	trump$_{10}$	frontera$_{11}$	epn$_{12}$
usa$_{13}$	migrante$_{14}$	amlo$_{15}$	

Table 2. Immigration dataset from July 11th of 2016 to July 11th of 2019

Social Media	**Language**	**Feature**	**Value**
X (Twitter)	English	Number of properties per tweet	71
		Number of tweets	4,412,621
		Number of geolocated tweets	1,755,468
		Avg. tweets per day	19,296.35
		Avg. geolocated tweets per day	8,471.6
		Avg. words per tweet	17.64
		Avg. file size per day	87.71 MB
	Spanish	Number of properties per tweet	71
		Number of tweets	1,974,944
		Number of geolocated tweets	732,835
		Avg. tweets per day	2,466.25
		Avg. geolocated tweets per day	1,522.67
		Avg. words per tweet	21.96
		Avg. file size per day	10.13 MB
	Number of tweets (both languages)		6,387,565
Reddit	English	Number of properties per post	15
		Number of posts	2,563,812
		Avg. posts per day	2501.98
		Avg. words per post	59.26
		Avg. file size per day	3.15 MB
Total number of texts about immigration			**8,951,377**

faster and simple; Healthcare systems [11], where distinct tools collect massive amounts of medical information for detecting critical insight of patients; and Cyber-security [15], where spam filtering and automatic intruder detection tools are making possible to identify malicious users. Text mining techniques are becoming more integrated and easier to use on the web.

This in turn, have introduce many approaches related to trending elements. Examples of this include: the analysis of digital marketing [39], recommendation systems [4], decision-making [36] and social network analysis [19]. For this last one, the extraction of non-trivial knowledge related to what is trending and whats is not have made an special effect on how users understand and consume information.

Considering its impact. it can be seen the importance of text mining and how it help others to make the most of their data, which leads to better decisions. Without this kind of tools, it would be impossible to analyze massive amounts of information (mainly on the web) which in consequence will stop the growing-flow of knowledge on the web.

2.2 Text Mining on Political Documents

In the context of social media (X, Facebook, Reddit, etc.), the analysis of political data by text mining techniques [22] have gained momentum considering the large amount of textual information, the number of interactions and the importance of the topic today (specifically on the U.S.) [29].

Citizenship and law enforcement topics have different text mining approaches that deal with the extraction of relevant users using social network measures (closeness, degree centrality, etc.) [31, 14]. Other approaches [32] used different ML algorithms (triplets and clustering) for detecting communities that might be important in a specific context. Other kind of approaches use classic statistic metrics like mean, median, mode, etc combined with IR scrapers for detecting structural patterns on social media data [18].

Immigration and border Security topics [7] also have distinct text mining approaches like the use of a friend of a friend (foaf) and co-occurrence graphs for obtaining activity patterns associated to users [17] or for summarizing/understanding large amounts of textual information [33]. Other kind of techniques rely on NLP for applying Part of Speech (PoS tags) and entity recognition techniques for scrapping linguistic features that help to understand the structure and purpose

Table 3. Immigration statistics: Baseline analysis

Statistic metric	Result	Description
Average number of words in X and Reddit	X: 16.81 Reddit: 87.62	Amount of words found in texts.
Average number of phrases in X and Reddit	X: 9.52 Reddit: 47.13	Groups of words that form meaningful units within a sentence.
Average number of sentences in X and Reddit	X: 5.52 Reddit: 26.13	Groups of words that make complete ideas.
Average Word Length	X: 6.34 Reddit: 4.76	Used to see how usual/unusual are the words in texts.
Number of different words on English language	X: 1,320,491 Reddit: 128,544	Distinct English words used in the immigration text documents.
Number of different words on Spanish language	X: 211,694 Reddit: 0	Distinct Spanish words used in the immigration text documents.
How users start a sentence	X: RT Reddit: What	X: Most of texts are a reply of other users. Reddit: Most of posts are questions related to immigration.
How users end a sentence	X: URL Reddit: ?	X: Most of texts finish with a reference to the source of the information. Reddit: Most of posts finish with the question mark, reinforcing the theory that most Reddit posts are questions
Most frequent PoS tags on X	Nouns Adjectives Verbs	References (nouns) to persons, places, things, or ideas are the most frequent words.
Most frequent PoS Tags on Reddit	Adjectives Nouns Adverbs	Words (adjectives) that describe or clarify nouns are the most frequent words.
Number of different users (@) on X	English: 68,447 Spanish: 13,671	Diversity of users that talk about immigration on X.
Number of different subtopics (#) on X	English: 8,590 Spanish: 1642	Diversity of subtopics related to immigration on X.
Number of different web sources (URLs) on X	English: 448,420 Spanish: 35,546	Diversity of URLs related to immigration on X.
Number of different web sources (URLs) on Reddit	12,674	Diversity of URLs related to immigration on Reddit.
Text documents distribution	X Eng: 79.7% X Spa: 10.2% Reddit Eng: 10.1%	Percentage of text documents in the dataset.
Geolocated Text documents distribution on X	English: 82.1% Spanish: 17.9%	Percentage of Geolocated documents in the dataset.

Table 4. Immigration statistics: Most/less frequent words

Frequent words on the dataset (X and Reddit)			
Ranking	**X English**	**X Spanish**	**Reddit English**
1	rt	rt	visa
2	immigration	pasaporte	greencard
3	trump	paisano	usa
4	passport	mojado	immigration
5	nafta	migrante	work
Unusual words on the dataset (X and Reddit)			
Ranking	**X English**	**X Spanish**	**Reddit English**
1	brown-skinned	vacaciones	season
2	neighbourhood	tramite	paperwork
3	muslim	preguntas	article
4	traveling	tramite	reasons
5	Rusia	ilegalidad	telephone
Most frequent users and topics on X			
Ranking	**X users (@)**	**X topics (#)**	
1	realDonaldTrump	immigration	
2	BreitbartNews	Trump	
3	HillaryClinton	MAGA	
4	FoxNews	migration	
5	FAIRImmigration	ny	
Less frequent users and topics on X			
Ranking	**X users (@)**	**X topics (#)**	
1	SarahPinder2	CNNIdiots	
2	tatianashanks	RefugeeRights	
3	ErfanSoomro	Illuminati	
4	HoopsmanB	yeahRight	
5	RamblinGrimace	FeelingSad	

of information (attitude, entities, sentiments etc.) [20]. Other approaches [6] have deal with the immigration analysis by detecting geo-spatial patterns of users on the Mexico-U.S. border. Additionally, distinct efforts [2] have implemented mining techniques for automatic content-characterization of news, stories and blogs related to the immigration phenomena.

From the different mining approaches implemented for political purposes, it can be seen that is a growing area where more interactions are available everyday. This in turn have made possible to analyze relevant text patterns from different social media sources, which applied in different research and decision-making process have a meaningful impact on how users understand, digest and use knowledge.

3 Text Mining Process

Data mining [41] and text mining are similar in terms of the way they extract valuable insight from data. The first one focus on the analysis of distinct data types (texts, images, sound, etc.) while the second one focuses only on the retrieval of textual information. Despite this key difference, both mining approaches have similar steps involved in the knowledge discovery process with the exception that in text mining, special emphasis is made on the data modeling considering the unstructured nature of texts and the different linguistic aspects to explore. Taking that in mind, Figure 1 shows the proposed steps to retrieve text patterns on the context of the U.S. immigration phenomena. The approach consists of three overall steps:

1. Dataset creation (see Section 4)
   - 1.1 Text acquisition: Create a large collection of text documents using different social media resources: X, Reddit, etc.
   - 1.2 Text preprocessing: Preprocess documents to guarantee homogeneity among texts.
2. Statistical analysis: (see Section 5)
   - 2.1 Choose distinct statistical metrics depending of the specifics of text documents.
   - 2.2 Extract statistical information of texts that describes and explores the nature of the data on the underlying topic.
3. Graph analysis: (see Section 6)
   - 3.1 Create rich representations for the text documents (see Sections 6.1 and 6.2).
   - 3.2 Uncover insightful knowledge from data representations using social network analysis tools.

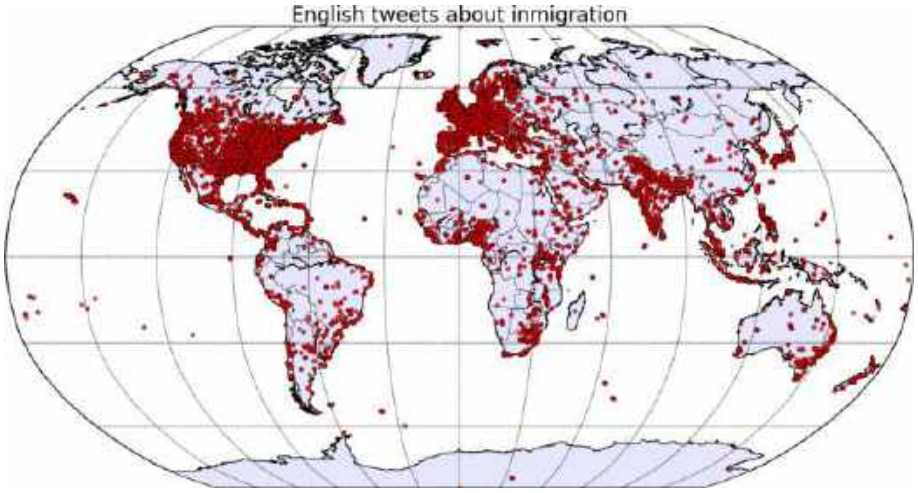


Fig. 2. Immigration statistics: English language texts distribution on X

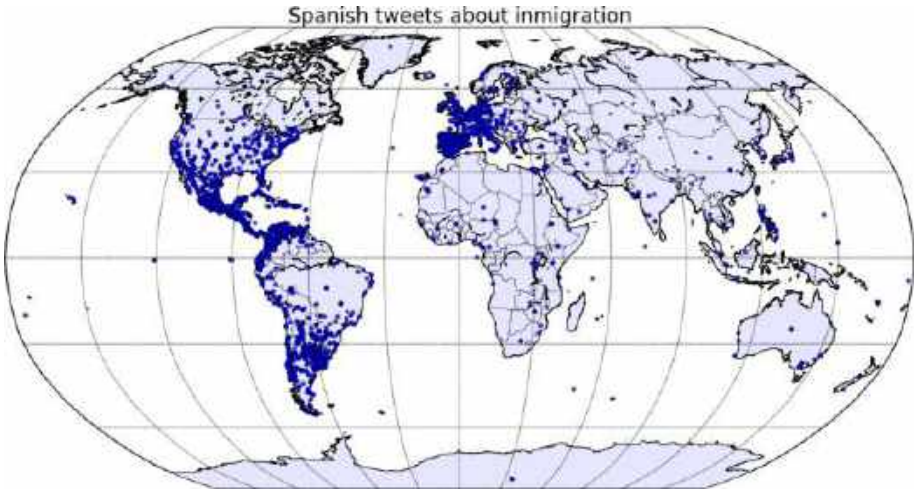


Fig. 3. Immigration statistics: Spanish language texts distribution on X

3.3 Use the obtained information to characterized in a better way a trending topic (see Sections 6.4 and 6.5).

From the previous figure, the steps associated to the analysis of the immigration phenomena are presented. The first step deals with the information gathering from distinct social media channels, making special emphasis on text acquisition and cleaning. The second step deals with the empirical analysis and inference of knowledge by using classic statistical metrics.

Finally, the third step involves the use of robust data structures (graphs) and social network analysis tools to uncover relevant patterns that statistics are unable to discover.

4 Dataset Creation

In this section, the collection of text documents associated to the immigration topic on English and Spanish languages is discussed. First, the keywords used for extracting text samples from

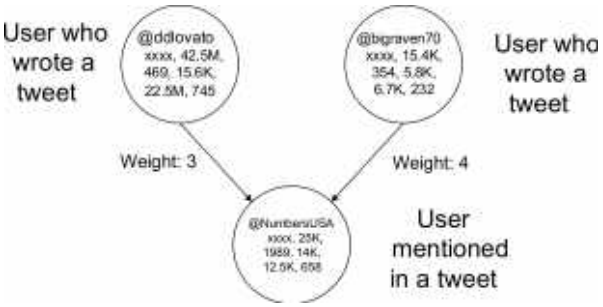


Fig. 4. User interaction graph example

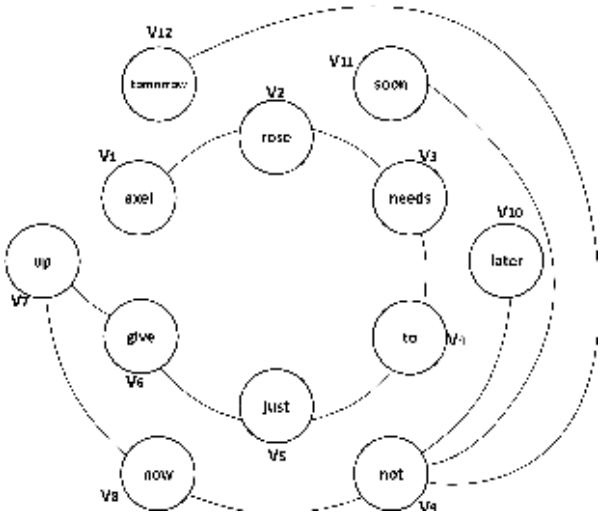


Fig. 5. Example of a co-occurrence graph with a window of two words

social media through different APIs (Application Programming Interfaces) are shown. Later, the description of the chosen social media sources used and the type of text documents obtained are presented. Finally, the main dataset features associated to the immigration topic are displayed considering the proposed text mining method (see Section 3).

4.1 Keyword Selection

In order to obtain suitable information from social media, words related to the immigration topic on July 2016 (dataset starting point) were extracted according to their frequency of occurrence on one thousand web pages[1]. For each language, the fifteen keywords without stopwords[2] or special characters are extracted from the web pages. These words were used as input for diverse social media APIs to obtain relevant text documents. Table 1 summarize words used on each language.

4.2 Social Media Sources

There are several social media APIs to obtain textual information, but most of them do not offer a public streaming to download texts periodically. Among the media channels that do not have this kind of restriction, X (formerly known as Twitter)[3] and Reddit[4] provide robust tools to download streaming data with minimum authentication and authorization from users over different languages.

Additionally, both media channels have real time interaction among users and cover a vast amount of domains around the world (U.S. specifically). Considering the previous features and the free availability of data on both social media channels, it was decided to use X and Reddit as primary data sources for detecting valuable knowledge related to the Immigration topic.

4.3 Text Documents Obtained

Using X and Reddit APIs, a dataset that comprise English and Spanish samples from U.S. users were collected. The dataset contains text documents from July 11 2016 to July 11 2019. All dataset samples were collected in a daily basis using a JSON format[5] with a UTF-8 encoding for storage each text document. Each collected sample contains properties/metadata related to the textual interaction. In the case of X[6], seventy one properties are obtained for each sample including: name, country, date, number of followers, number of likes, etc.

[1] The webpage number was decided based on the Google ranking and the amount of textual information.
[2] Stopwords represents a group of words that bear no content or relevant semantics in the text.
[3] twitter.com/
[4] www.reddit.com/
[5] JSON is a syntax for storing and exchanging data o the web.
[6] dev.twitter.com/overview/api/tweets

Table 5. Immigration graphs: main properties

Social Media	Graph Type	Vertex Type	Number of Vertices	Edge Type	Number of Edges
X and Reddit in English language	Co-occurrence	Words	1,380,615	Two words appear together in the text.	690,843
X and Reddit in Spanish language	Co-occurrence	Words	40,386	Two words appear together in the text.	90,128
X in English language	User interaction	Users	97,582	One user reference other in the text.	135,295

For Reddit[7], fifteen properties are collected including: author name, text date, country, topic name, topic score, subreddit name, etc. The dataset also provides some geolocation properties. In the case of X, this provides coordinates and place elements (if these are release by the user) while for Reddit, this does not provide any kind of metadata that can be used for obtaining the latitude and longitude associated to a post due to some API restrictions.

Table 2 summarize the dataset main features emphasizing the number of documents for Spanish and English languages on X and Reddit. From the dataset table, it can be observed that the English subset considered both social media channels while for Spanish it is only used X.

This is due to the lack of Spanish samples and the small amount of texts retrieve from the Reddit API. Additionally, The number of samples collected highlight the amount of interactions in both social media channels. In the case of X, there are more small interactions (140 characters at most) while for Reddit there are less interactions but these condensate more textual information.

5 Statistical Analysis

As a first attempt to obtain valuable knowledge from text documents, a statistical analysis [35] was applied according to the proposed approach (see Section 3).

The main goal of this phase, was to find and summarize the presence of textual patterns to describe or estimate information that can be helpful to understand the nature of the topic in the context of social media. Table 3 display some baseline statistics found in both media channels. For each entry, the metric used, the result obtained and a brief description is presented. From the previous table, it can be noticed that baseline statistics provide valuable insight about the data collected.

The average number of words, phrases and sentences indicate how diverse is the vocabulary as well as the amount of ideas/sentences used to talk about immigration. How users start or end a text also illustrate the kind of interactions associated to the topic (is a question or a reply from other users). Additionally, The analysis of PoS tags[8] permits to understand about whats people focus more on textual interactions (entities-ideas or descriptive elements about the topic).

Other statistical elements can be seen in Table 4 where frequent and unusual words from X and Reddit are shown. The table display frequent users (@) and topics (#) obtained from X interactions, this information is useful for understanding what is trending and what is not considering the frequency of occurrence of immigration words. In the case of geolocated texts on X, Figures 2 and 3 show a statistical tweet distribution on English and Spanish languages.

[7] www.reddit.com/dev/api/

[8] POS tagging is the process of marking up a word based on the syntactic role that it plays in a sentence.

Table 6. User interaction graph: Top X users on English and Spanish

Centrality Measure	English X users (@)		
Degree	$RealDonaldTrump_1$	CNN_2	$FoxNews_3$
	$HillaryClinton_4$	$YouTube_5$	$BarackObama_6$
Closeness	$RealDonaldTrump_1$	$FoxNews_2$	CNN_3
	$Nytimes_4$	$CBCNews_5$	$NumbersUSA_6$
Betweenness	$RealDonaldTrump_1$	$HillaryClinton_2$	$CBCNew_3$
	$MSNBC_4$	$YouTube_5$	$Reuters_6$
Centrality Measure	**Spanish X users (@)**		
Degree	EPN_1	$lopezobrado_2$	$GenPenaloza_3$
	$lopezdoriga_4$	$SREmx_5$	$YouTube_6$
Closeness	$RedsocialSAIME_1$	$LeonKrauze_2$	SRE_3
	$ElNacionalWeb_4$	$elpais_5$	$YouTube_6$
Betweenness	$YouTube_1$	$CNNEE_2$	$Telemundo_3$
	$Univision_4$	$TwitterEspanol_5$	$ElUniversal_6$

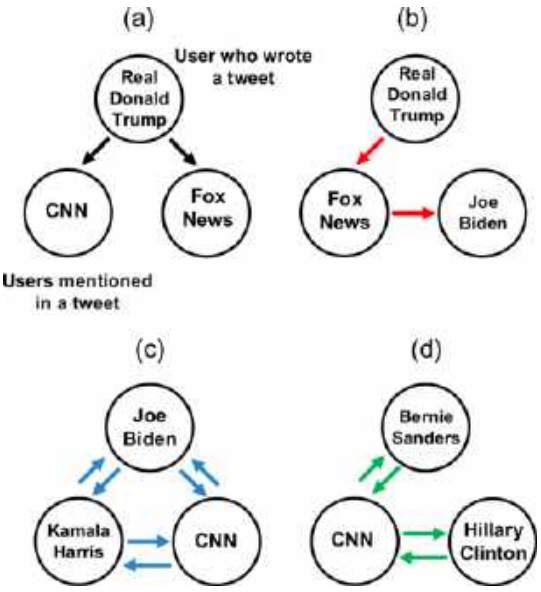


Fig. 6. User interaction graph: English triad examples found

In these figures, it can be observed that in the case of the English tweets most of U.S. users that talk about immigration were located (at time of posting) in North America, Europe and India. For the Spanish tweet distribution, the U.S. users that talk more about immigration were located in America (North America, Central America and South America) and Europe.

Taking into account some examples of statistics obtained from the dataset, it can be remarked the importance of this kind of elements for a basic understanding of the immigration phenomena. The analysis of frequent elements on X and Reddit provide an intuitive way for detecting relevant words, users, topics, etc. This in turn, provide insightful knowledge for implementing more advance data representations (graphs).

6 Graph Analysis

In this section are described the patterns obtained using more complex structures like graphs in the context of a text mining approach (see Section 3). First, two graph representations are proposed: one based on the interaction of words related to immigration (word co-occurrence) and another based on the interaction of users that talk about the topic. Latter, the patterns/subgraphs obtained are presented and described.

6.1 User Interaction Graph

Taking into consideration how users relate to each other and the importance of this relationships to understand social media synergy, a graph that represents the interaction among users that talk about immigration topics on X is proposed (Reddit does not give much importance to users as X does). Formally proposed graph $G = (V, E, L_V, L_E)$ has the following attributes:

1. $V = \{v_1, ..., v_n\}$ is a finite set of vertices that consists of the X users (@) contained in one or several texts.
2. $E \subseteq V \times V$ is the finite set of edges which represent that a user referenced other user or that other user referenced him in another text.
3. L_V is the label set of V, where each vertex has the following properties:
   - X user name: Name of the X user (@).
   - X user ID: Unique identifier for this text.
   - Followers count: The number of followers this account currently has.

Fig. 7. User interaction graph: Community of Spanish users found

- Friends count: The number of users this account is following (AKA their ”followings”).
- Statuses count: The number of text (including re-texts) issued by the user.
- Favorites count: Indicates approximately how many times a text has been ”liked” by X users.
- Listed count: The number of public lists that this user is a member of.

4. L_E is the label/weight set of E, where: L_E = #hashtags + #URLs + #interactions that have in common two X users (two vertices).

As an example of this graph-based representation, consider the following texts extracted from two texts from X:

- **User1**: @ddlovato, **Text1**: @NumbersUSA #ElectionDay #IVotedBecause I believe in equality, we need comprehensive immigration.
- **User2**: @bigraven70, **Text2**: @NumbersUSA I'm all for legal immigration #immigration. My family came here legally and yours can too #Trump URL.

Based on proposed properties, user and text information previously preprocessed[9] can be mapped to the user interaction graph shown in Figure 4. In the user interaction graph, the communication (edges) between X users (vertices) that talk about the immigration in English is considered.

The goal of this graph, is to take advantage of the properties provided by X (and not by Reddit) like the users (@), topics (#) and even the web resources (URLs) to propose a weighting scheme that could be used to uncover meaningful users, communities and new interactions. This graph can be seen as a special foaf graph[10] [38] where the relationships are oriented to model how users relate to each other when they are mentioned in a specific context.

6.2 Co-Occurrence Graph

Keeping in mind the relevance of lexical elements (based on previous section) and the lack of syntactic structure on social media documents. A non-directed and unweighted graph representation based on the co-occurrence [31, 40] of two words is proposed.

The objective of this graph is to take advantage of all natural interactions between words[11] to extract valuable lexical-syntactical patterns that can not be obtain using traditional statistics. Formally, the proposed co-occurrence graph used in the experiments is represented by $G = (V, E, L_V)$,where:

1. $V = \{v_1, ..., v_n\}$ is a finite set of vertices that consists of the words contained in many texts.
2. $E \subseteq V \times V$ is the finite set of edges which represent that two vertices are connected if their corresponding lexical units co-occur within a window of two words in the text at least once.
3. L_V is the label set of V, where $L_V = \{\text{etq} : \text{etq} \in \text{words}\}$

As an example, consider the following sentence ζ extracted from a text T: “Axel Rose needs to just give up. Now. Not later, not soon, not tomorrow.”,

[9] Something similar to the co-occurrence graph.

[10] Friend of a friend graph structure.

[11] The bond of one term over another one in the syntactic order.

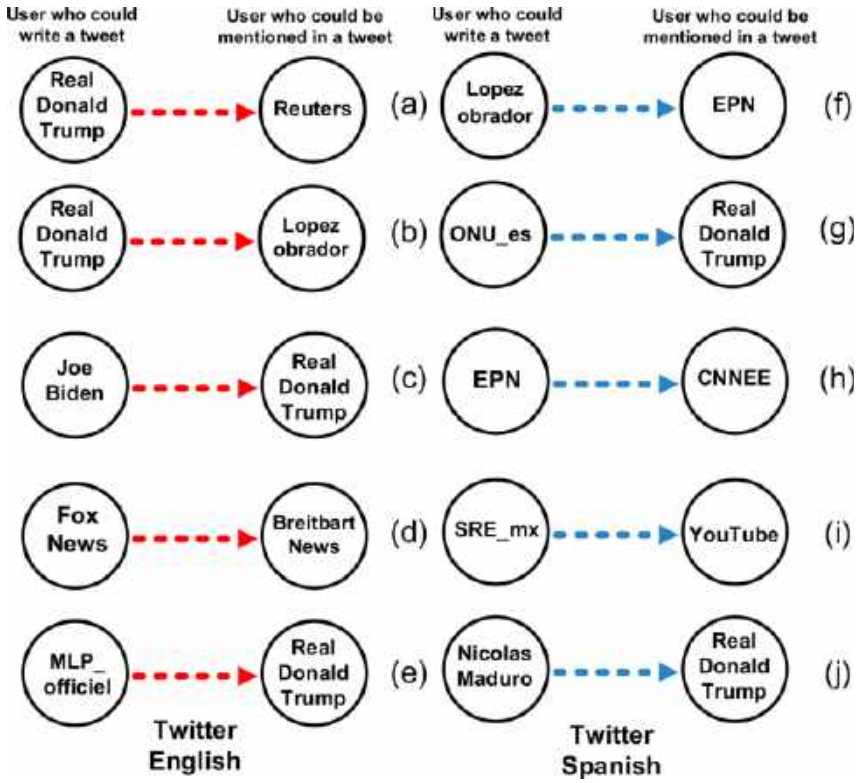


Fig. 8. User interaction graph: Link prediction examples found

which after the preprocessing stage[12] would be as follows: “axel rose needs to just give up now not later not soon not tomorrow”. Based on the proposed representation, preprocessed sentence ζ can be mapped to the co-occurrence graph shown in Figure 5.

The proposed co-occurrence graph captures the syntactic interactions (edges) of all the words (vertices) used when people discussed about immigration in X-Reddit (for English and Spanish languages). The idea of this graph is to obtain relevant words, phrases and even sentences that describe the way individuals write about the topic for understanding attitudes, trends and media consumption.

[12]This task includes lowercase all words in the texts and elimination of punctuation symbols that are not part of the ASCII encoding (except for @, # and URLs in the case of X).

6.3 Graph Properties

In order to understand the richness of the two proposed graph structures (see Sections 6.1 and 6.2). Table 5 shows main features associated to the co-occurrence graph and the user interaction graph in the context of the social media sites analysed. From table 5, it can be observed the amount of vertices and edges created from the social media dataset. This highlight the diversity of information obtained trough several months as well as the complexity of the data structures created. Considering graph properties, upcoming sections show results examples obtained by optimized graph mining algorithms [30].

6.4 User Interaction Graph Results

As in the statistical phase, the main objective of the user interaction graph is to obtain insightful knowledge from social media.

Table 7. Co-occurrence graph: Top words on X and Reddit

Centrality Measure	**English Language Words**			
Degree	RT_1	$immigration_2$	$Trump_3$	$passport_4$
	USA_5	$migration_6$	$people_7$	$visa_8$
Closeness	$immigration_1$	$Trump_2$	$election_3$	$Clinton_4$
	$Illegal_5$	$passport_6$	$Obama_7$	$Jobs_8$
Centrality Measure	**Spanish Language Words**			
Degree	RT_1	$pasaporte_2$	$paisano_3$	$indocumentado_4$
	$migrante_5$	$EEUU_6$	$frontera_7$	$Europa_8$
Closeness	$visa_1$	$trabajo_2$	$drogas_3$	$migrante_4$
	$pasaporte_5$	$frontera_6$	$remesa_7$	$Trump_8$

In this case, there are extract relevant users from X based on distinct social network analysis metrics (centrality, community detection and link prediction). Table 6 show top X users in English and Spanish languages using a reinterpretation of a classic social network analysis called centrality measures[13] [9]:

- **Degree centrality**: Collect users who are more referenced on immigration texts and users who referenced many others about the topic.
- **Closeness centrality**: Obtain users who are referenced immediately about immigration and users who spread out information faster about the topic.
- **Betweenness centrality**: Select people who have access to a lot of users that talk about immigration.

Other elements retrieved from the user interaction graph are triad elements [24]. This provide topological insight of the interaction of three interconnected vertices. These kind of subgraphs helps to understand how users spread information about immigration and how close they are in terms of their interactions.

[13] Graph Centrality refers to a family of structural measures related to the position/importance of vertex in a graph representation.

The triads with highest edge weight are consider as relevant communities that maintain a constant communication flow. Figure 6 display some examples of triads found on English language (something similar is performed for Spanish language). In addition to the triad detection, the uncovering of bigger communities is also performed by the user interaction graph.

The edge-betweenness centrality [37] is used for detecting high interconnected vertices on the network. The idea of this technique, is to gradually remove the weighted edges with highest betweenness and recalculate the centrality for all edges after every removal. This way sooner or later the network falls off into smaller components which are relevant user groups that talk about immigration.

Figure 7 illustrates the types of user communities obtained on the context of the immigration topic for the Spanish language (something similar is performed for English language). Finally, there is a applied an link prediction approach [28, 8] for inferring new relationships among users based on the topological structure of the graph and the weighted scheme proposed.

This kind of technique analyzed the neighborhood of vertices on the graph. Vertices that have similar neighbors will have new edges in the near future while vertices that do not share

Table 8. Co-occurrence graph: Top phrases on X and Reddit

Centrality Measure	**English Language Words**			
Degree	RT_1	$immigration_2$	$Trump_3$	$passport_4$
	USA_5	$migration_6$	$people_7$	$visa_8$
Closeness	$immigration_1$	$Trump_2$	$election_3$	$Clinton_4$
	$Illegal_5$	$passport_6$	$Obama_7$	$Jobs_8$
Centrality Measure	**Spanish Language Words**			
Degree	RT_1	$pasaporte_2$	$paisano_3$	$indocumentado_4$
	$migrante_5$	$EEUU_6$	$frontera_7$	$Europa_8$
Closeness	$visa_1$	$trabajo_2$	$drogas_3$	$migrante_4$
	$pasaporte_5$	$frontera_6$	$remesa_7$	$Trump_8$

many neighbors will remain without change. Figure 8 show some examples of interactions inferred using the link prediction and the user interaction graph on the English language.

6.5 Co-Occurrence Graph Results

The use of a co-occurrence graph permit to obtain insightful knowledge from texts that comes from different social media. The co-occurrence representation allow to extract more complex patterns that do not depend entirely of the frequency of occurrence of texts (like in statistics). Tables 7, 8 and 9 show relevant examples of words, phrases and sentences obtained by using the co-occurrence representation and some of centrality measures[14]. The following interpretations are proposed for extracting elements from the graph:

- **Words**: Top ranked words by degree and closeness centralities are extracted considering that such elements could be highly mentioned in the syntactic structure of immigration texts.
- **Small phrases**: Words that have a high interaction with other words, regardless of their syntactic relevance in the texts are suitable candidates to obtain collocations[15].

So, the top ranked vertices/words according to the degree centrality that are part of a collocation are obtained.

- **Sentences**: Considering that vertices with a high closeness centrality are words with an important role in the syntactic sequence of texts (they are reachable in the minimum number of steps), the sentences that have most of top ranked words according to this centrality are extracted.

In table 7, top words are presented without considering stopwords and special symbols. From the words obtained, it can be observed that degree centrality obtained words that people mentioned immediately when the immigration topic is discussed, while in the closeness centrality, the words that play an active and central role in the texts related to immigration are extracted.

In the case of the top phrases related to immigration, Table 8 shows the top collocations with stopwords but not special symbols. Analyzing the collocations obtained using the degree centrality it is possible to see the relevant subtopics used when individuals write about immigration on X and Reddit.

For top sentences related to immigration, Table 9 show some examples that reflect the most important attitudes and ideas related to immigration. These examples are formed of words with high closeness elements which implies that these sentences could be used as a brief summary of what people write about immigration.

[14] Among different centrality elements, degree and closeness demonstrate to be stable in the previous section.

[15] Pairs of words that always appear together in the texts.

Table 9. Co-occurrence graph: Top sentences on X and Reddit

Centrality Measure	English Language Sentences
Closeness	1. Your vote will make a difference. #vote
	2. I voted for Trump because illegal immigration is Illegal
	3. The Era Of Climate Migration Meets Violent Borders
	4. We are more concerned about immigration than any other nation
Centrality Measure	**Spanish Language Sentences**
Closeness	1. Presidencia de Trump despierta temores entre indocumentados
	2. No podemos esperar nada bueno de biden y kamala.
	3. ¿SERA? Mexico, preparado ante deportación de mexicanos
	4. Cancelación de la #deportacion de #EstadosUnidos?

7 Conclusions and Future Work

An approach that implements a text mining method based on statistical and graph analysis has been presented. Results obtained highlight the relevance of the implemented method and the importance of extracted patterns (statistical and graph ones) for explaining the nature of the immigration topic. Considering the theoretical implications of this mining method, the practical benefits associated are the following:

– The analysis of classic statistical metrics (frequency mainly) provide initial insight associated to immigration topic, and supply information for creating the graph-based representations proposed (like which centrality measure used considering the frequencies found).
– For mining the immigration topic, the user interaction graph and the co-occurrence graph with a window of two words show to be a very effective option to extract important information of texts, revealing that the co-occurrence graph not only works for classic NLP problems [10] but also for extracting distinct types of lexical/syntactical patterns on social media.

In the case of the user interaction graph, this showed to be a really good option to understand the dynamics of users in a specific network but it is necessary to test this kind of graph in other trending topics.

– The use of co-occurrence windows of two words allows to map the natural relationship of terms, which facilitate the analysis of lexical and syntactical elements of texts related to immigration.
– The co-occurrence graph can be used to extract topologically important words, phrases and even complete sentences that represent what people think or express when they write about the immigration topic.
– The user interaction graph permits to map the way in which X users relate to each other in the context of a specific topic, instead of just mapping the static relations that users have with others on a social network like in the classic friend of a friend graph.
– A weighted scheme on the interaction graph permits to evaluate in a more accurate way the user communication, considering the topics in common, the URLs and the frequency of occurrence between vertices.

This schema ultimately leads some graph mining algorithms (triad and community detection as well as link prediction) to find strong relationships between users that have multiple elements in common when they write about immigration.

– One of the major differences between the co-occurrence graph and the user interaction graph is that the co-occurrence of words can be applied to any kind of text document as long as the texts have a known encoding (like UTF-8 or ASCII). So, co-occurrence graphs offer more flexibility because they can be used in any text in any language with minimum preprocessing while the interaction graph permit an accurate analysis of X due to the social media specific properties.

Research on the use of a text mining approach continues in favor of improving obtained findings, keeping in mind the complexity of the use of graphs. Ongoing and future work includes the following actions:

– Work with experts on the U.S immigration problem to identify the impact of extracted patterns over decision making.
– Extract new features from graphs associated to the distinct levels of language to improve previous results [34].
– Experiment with other graph-based representations for documents that include semantic information related to texts [26].
– Applying different visualization methods on graph structures to present and understand obtained textual information in a more natural and easy-to-understand manner [25, 16].
– Analyze other trending topics like healthcare, news diffusion, etc. [1, 12], for testing the behavior of proposed graphs when applied to other real-world text documents.

Acknowledgments

This work has been partially supported by the CONACYT grant with reference #373269/244898. The authors would also like to thank Darnes Vilariño Ayala and David Báez López for their invaluable help reviewing this manuscript.

References

1. **Ahn, S. J., Yoon, H. Y., Lee, Y. J. (2021).** Text mining as a tool for real-time technology assessment: Application to the cross-national comparative study on artificial organ technology. Technology in Society, Vol. 66, pp. 101659. DOI: 10.1016/j.techsoc.2021.101659.
2. **Altarrazi, S. M., Sasi, S. (2016).** Tweeple's microblogs on illegal immigration in USA. International Conference on Electrical, Electronics, and Optimization Techniques, pp. 2011–2018. DOI: 10.1109/iceeot.2016.7755041.
3. **Balaji, T. K., Rao-Annavarapu, C. S., Bablani, A. (2021).** Machine learning algorithms for social media analysis: A survey. Computer Science Review, Vol. 40, pp. 100395. DOI: 10.1016/j.cosrev.2021.100395.
4. **Betancourt, Y., Ilarri, S. (2020).** Use of text mining techniques for recommender systems. Proceedings of the 22nd International Conference on Enterprise Information Systems, pp. 780–787. DOI: 10.5220/0009576507800787.
5. **Biemann, C., Mehler, A. (2014).** Text mining: From ontology learning to automated text processing applications. Springer International Publishing. DOI: 10.1007/978-3-319-12655-5.
6. **Borruso, G. (2009).** Geographical analysis of foreign immigration and spatial patterns in urban areas: Density estimation and spatial segregation. Lecture Notes in Computer Science, Vol. 5072, pp. 459–474. DOI: 10.1007/978-3-540-69839-5_34.
7. **Cartwright, K., Chacon, L. (2021).** The impact of immigration-related separation and reunification on children's education: Evidence from the american community survey 2010–2018. Children and Youth

Services Review, Vol. 126, pp. 106013. DOI: 10.1016/j.childyouth.2021.106013.

8. **Castillo, E., Cervantes, O., Vilariño, D. (2018).** Author profiling using a graph enrichment approach. Journal of Intelligent and Fuzzy Systems, Vol. 34, No. 5, pp. 3003–3014. DOI: 10.3233/jifs-169485.
9. **Castillo, E., Cervantes, O., Vilariño, D. (2019).** Authorship verification using a graph knowledge discovery approach. Journal of Intelligent and Fuzzy Systems, Vol. 36, No. 6, pp. 6075–6087. DOI: 10.3233/jifs-181934.
10. **Castillo-Juarez, E., Cervantes-Villagómez, O., Vilariño-Ayala, D. (2018).** Text analysis using different graph-based representations. Computación y Sistemas, Vol. 21, No. 4. DOI: 10.13053/cys-21-4-2551.
11. **Chatterjee, S., Goyal, D., Prakash, A., Sharma, J. (2021).** Exploring healthcare/health-product ecommerce satisfaction: A text mining and machine learning application. Journal of Business Research, Vol. 131, pp. 815–825. DOI: 10.1016/j.jbusres.2020.10.043.
12. **Chen, W. K., Chen, L. S., Pan, Y. T. (2021).** A text mining-based framework to discover the important factors in text reviews for predicting the views of live streaming. Applied Soft Computing, Vol. 111, pp. 107704. DOI: 10.1016/j.asoc.2021.107704.
13. **Coenen, F., Fred, A., Aveiro, D., Dietz, J., Bernardino, J., Masciari, E., Filipe, J. (2023).** Knowledge discovery, knowledge engineering and knowledge management. 14th International Joint Conference. Springer Cham. DOI: 10.1007/978-3-031-43471-6.
14. **Cook, D. J., Holder, L. B. (2006).** Mining graph data. John Wiley and Sons.
15. **de-Boer, M. H. T., Bakker, B. J., Boertjes, E., Wilmer, M., Raaijmakers, S., van-der-Kleij, R. (2019).** Text mining in cybersecurity: Exploring threats and opportunities. Multimodal Technologies and Interaction, Vol. 3, No. 3, pp. 62. DOI: 10.3390/mti3030062.
16. **Evergreen, S. (2016).** Effective data visualization: The right chart for the right data. SAGE Publications.
17. **Fotouhi, B., Rabbat, M. G. (2012).** Migration in a small world: A network approach to modeling immigration processes. Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing, pp. 136–143. DOI: 10.1109/allerton.2012.6483210.
18. **Freire-Vidal, Y., Graells-Garrido, E. (2019).** Characterization of local attitudes toward immigration using social media. Companion Proceedings of The 2019 World Wide Web Conference, pp. 783–790. DOI: 10.1145/3308560.3316455.
19. **Gao, W., Sebastiani, F. (2015).** Tweet sentiment: From classification to quantification. Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 97–104. DOI: 10.1145/2808797.2809327.
20. **Huber, P., Oberdabernig, D. A. (2016).** The impact of welfare benefits on natives' and immigrants' attitudes toward immigration. European Journal of Political Economy, Vol. 44, pp. 53–78. DOI: 10.1016/j.ejpoleco.2016.05.003.
21. **Ignatow, G., Mihalcea, R. (2017).** An introduction to text mining: Research design, data collection, and analysis. SAGE Publications, Inc.
22. **Ignatow, G., Mihalcea, R. (2017).** Text mining: A guidebook for the social sciences. SAGE Publications, Inc. DOI: 10.4135/9781483399782.
23. **Jeong, Y., Suk, J., Hong, J., Kim, D., Kim, K. O., Hwang, H. (2018).** Text mining of online news and social data about chatbot service. Communications in Computer and Information Science, pp. 429–434. DOI: 10.1007/978-3-319-92270-6_61.
24. **Jia, S., Gao, L., Gao, Y., Nastos, J., Wen, X., Zhang, X., Wang, H. (2017).** Exploring

triad-rich substructures by graph-theoretic characterizations in complex networks. Physica A: Statistical Mechanics and its Applications, Vol. 468, pp. 53–69. DOI: 10.1016/j.physa.2016.10.021.

25. **Jonker, D., Brath, R. (2015).** Graph analysis and visualization: Discovering business opportunity in linked data. John Wiley and Sons.
26. **Kejriwal, M., Knoblock, C. A., Szekely, P. (2021).** Knowledge graphs: Fundamentals, techniques, and applications. The MIT Press.
27. **Kushwaha, A. K., Kar, A. K., Dwivedi, Y. K. (2021).** Applications of big data in emerging management disciplines: A literature review using text mining. International Journal of Information Management Data Insights, Vol. 1, No. 2, pp. 100017. DOI: 10.1016/j.jjimei.2021.100017.
28. **Liben-Nowell, D., Kleinberg, J. (2003).** The link prediction problem for social networks. Proceedings of the 12th International Conference on Information and Knowledge Management, pp. 556–559. DOI: 10.1145/956863.956972.
29. **Light, M. T., Thomas, J. T. (2021).** Undocumented immigration and terrorism: Is there a connection?. Social Science Research, Vol. 94, pp. 102512. DOI: 10.1016/j.ssresearch.2020.102512.
30. **Malak, M. S., East, R. (2016).** Spark GraphX in action. Manning Publications.
31. **Mihalcea, R., Radev, D. (2011).** Graph-based natural language processing and information retrieval. Cambridge University Press. DOI: 10.1017/cbo9780511976247.
32. **Miranker, M., Giordano, A. (2020).** Text mining and semantic triples: Spatial analyses of text in applied humanitarian forensic research. Digital Geography and Society, Vol. 1, pp. 100005. DOI: 10.1016/j.diggeo.2020.100005.
33. **Mukherjee, S., Oates, T., DiMascio, V., Jean, H., Ares, R., Widmark, D., Harder, J. (2020).** Immigration document classification and automated response generation. International Conference on Data Mining Workshops, pp. 782–789. DOI: 10.1109/icdmw51313.2020.00114.
34. **Negro, A. (2021).** Graph-powered machine learning. Manning Publications.
35. **Ott, R. L., Longnecker, M. T. (2015).** An introduction to statistical methods and data analysis. Cengage.
36. **Ozcan, S., Suloglu, M., Sakar, C. O., Chatufale, S. (2021).** Social media mining for ideation: Identification of sustainable solutions and opinions. Technovation, Vol. 107, pp. 102322. DOI: 10.1016/j.technovation.2021.102322.
37. **Raghavan, U. N., Albert, R., Kumara, S. (2007).** Near linear time algorithm to detect community structures in large-scale networks. Physical Review E, Vol. 76, No. 3. DOI: 10.1103/physreve.76.036106.
38. **Robinson, I., Webber, J., Eifrem, E. (2013).** Graph databases. O'Reilly Media, Inc.
39. **Saura, J. R. (2021).** Using data sciences in digital marketing: Framework, methods, and performance metrics. Journal of Innovation and Knowledge, Vol. 6, No. 2, pp. 92–102. DOI: 10.1016/j.jik.2020.08.001.
40. **Sonawane, S., Kulkarni, P. A. (2014).** Graph based representation and analysis of text document: A survey of techniques. International Journal of Computer Applications, Vol. 96, No. 19, pp. 1–8. DOI: 10.5120/16899-6972.
41. **Witten, I. H., Frank, E., Hall, M. A. (2011).** Data mining: Practical machine learning tools and techniques. Morgan Kaufmann Publishers Inc. DOI: 10.1016/c2009-0-19715-5.

Article received on 15/04/2023; accepted on 18/04/2024.
** Corresponding author is Esteban Castillo.*

ISSN 2007-9737

Multi-Class Sentiment Analysis of COVID-19 Tweets by Machine Learning and Deep Learning Approaches

Maaskri Moustafa[1,*], Sid Ahmed Mokhtar-Mostefaoui[1], Madani Hadj-Meghazi[1], Mohamed Goismi[2]

[1] University of Tiaret, LRIAS Laboratory, Computer Science Department, Algeria

[2] Dr.Tahar Moulay University GeCoDe Laboratory, Computer Science Department, Algeria

{moustafa.maaskri, h.meghazi}@univ-tiaret.dz, s_mostefaoui@esi.dz, mohamed.goismi@univ-sba.dz

Abstract. COVID-19 is a virus that has spread rapidly over the globe. The condition has repercussions beyond the realm of public health. Twitter is one platform where people post reactions to events during the outbreak. User-generated information, like tweets, presents unique challenges for sentiment analysis on Twitter data. With that in mind, this work employs four methods for analyzing Twitter data in terms of sentiment: the vector space model (TF-IDF) with three different ensemble machine learning models (voting, bagging, and stacking) and BERT (Bidirectional Encoder Representations from Transformers). Experiments showed that BERT outperformed the other three techniques, with an F1-score of 74%, a precision of 74%, and a recall of 74% for categorizing five sentiment classes on data from a Kaggle competition (Coronavirus tweets NLP-Text Classification).

Keywords. Ensemble machine learning, deep learning, voting, bagging, stacking, BERT.

1 Introduction

Several social media platforms are generating enormous volumes of text data these days, which has sparked a renewed interest in data processing to uncover the data's underlying meaning in a broader setting. Because Twitter data are accessible to the public and handled transparently, they may be used to investigate novel natural language processing (NLP) and data mining approaches, such as sentiment analysis [4]. The personal information, opinion, or polarity communicated in phrases or paragraphs may be extracted via sentiment analysis.

A valuable technique that offers real-time monitoring and decision-making capacities in the battle against the COVID-19 epidemic is sentiment analysis of data from social media platforms such as Twitter. This kind of analysis may be used to extract information from raw data. Many nations have implemented steps like isolation, quarantine, lockdown, or social distancing to address social media fears about the COVID-19 pandemic [13, 18].

However, different ethnicities and cultures have different methods of expressing their ideas. No matter the topic (health, politics, sports, or entertainment), people in one nation may react more passionately than others. Data-driven machine learning (ML) techniques predict [11, 23].

ML algorithms are widely utilized in health informatics [12, 5], pandemic predictions [13, 31], autism prediction [16], and many other fields. Many researchers have used ML systems to analyze Twitter sentiment. Villavicencio et al. [29] used

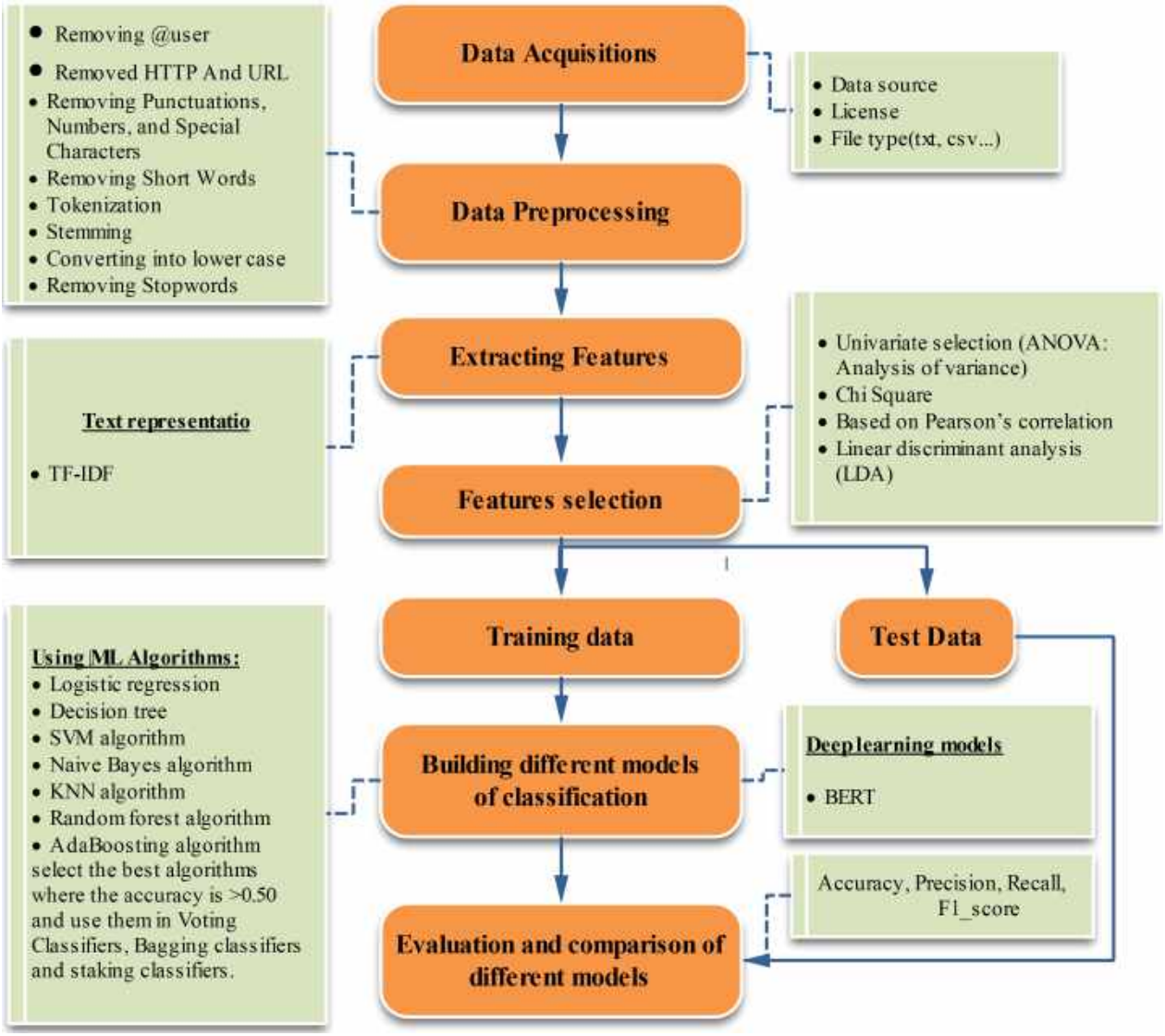


Fig. 1. Proposed scheme diagram

the Naïve Bayes classifier to analyze COVID-19 vaccination tweets in the Philippines and obtained 81.77% accuracy. The classifier was tested on 11,974 manually tagged tweets. Khan et al. [17] used the Naïve Bayes classifier to sentiment score 50,000 COVID-19 tweets and discovered 19% positive and 70% negative tweets. The authors of [15] employed deep learning classifiers to categorize 600 COVID-19-related tweets by sentiment. H-SVM had the most remarkable accuracy (86%), recall (69%), and F1-score (77%), among the classification methods employed in their research. Gupta et al. [9] investigated Twitter users' perceptions of the impact of weather on SARS-CoV-2 transmission.

The research filtered relevant tweets (n = 28555) using 11 ML algorithms and classified annotated tweets (n = 2442) into sentiment labels. The relevant tweet dataset showed 40.4% ambiguity regarding weather's influence, 33.5%

Table 1. Sample Tweet data for sentiment classification

OriginalTweet	Sentiment
The Home Depot is limiting the number of customers allowedinto its stores at any one time	Positive
I SERIOUSLY DOUBT anyone will be voting for ANY RepublicanPlease wear a mask take hand sanitizer and vote these bastards out	ExtremelyNegative
I thought I would save more money by being quarantined butonline shopping determined that was a lie. ???\r\r\n #CoronaCrisis	Extremely Positive

no effect, and 26.1% some effect on SARS-CoV-2 transmission. Latent Dirichlet Allocation (LDA) modeling was used to identify COVID-19-related topics from Twitter data [6, 1].

The researchers in [27] assessed machine learning classifiers on 7528 COVID-19 tweets. Automatic Twitter annotation yielded 93% accuracy in the trial. This research indicated that ML techniques were widely employed for COVID-19 tweet sentiment analysis and categorization. Due to the COVID-19 epidemic, no research has explicitly investigated ensemble ML models for sentiment analysis.

Nemez [22] employed a trained Recurrent Neural Network (RNN) to assess the percentage of positive, neutral, and negative attitudes in a coronavirus-related Twitter dataset. RNN forecasts showed 24.8% more positive tweets on May 13-14, 2020. Rustam [27] examined RF, XGBoost, SVC, ETC, DT, and LSTM for sentiment analysis. LSTM performed worse in that trial. The training data for the LSTM were insufficient.

Chakraborty [3] used a fuzzy approach using a Gaussian membership function to predict Twitter sentiment with 79% accuracy. According to particular research, sentiment analysis on Twitter data is difficult owing to the diversity, writing faults, and non-standard sentence patterns of user-generated information. This study analyzes COVID-19-related Twitter data using ensemble machine-learning methods and deep-learning models.

Voting, bagging, stacking, and BERT (Bidirectional Encoder Representations from Transformers) were tested for COVID-19 Twitter sentiment analysis. Coronavirus tweets' NLP-Text Classification Kaggle competition data already contains a sentiment class [20].

The following section presents the proposed scheme for sentiment analysis of COVID-19 tweets. Furthermore, Section 3 discusses the ML and deep learning model findings in detail. The final section concludes the article and highlights its limitations.

1.1 Proposed Scheme

The methodological overview of the sentiment analysis process is shown in figure 1. In the data accession step, the COVID-19-related tweets data were collected from Twitter. Moreover, the collected dataset was preprocessed, followed by word representation, classification methods, and performance measurement.

1.2 Data Acquisition

We have collected English-language tweets related to the coronavirus that were posted on Twitter between January 1, 2020, and December 31, 2020, sourced from several countries around the world through the pandemic timeline, and they are available at [20].

A set of predefined and widely used science and news media terms related to coronavirus, such as "COVID-19", "coronavirus", "lockdown", "isolation", "quarantine", "pandemic" and "ncov-2020" was used to collect tweets.

The data consisted of training data (41157) and testing data (3798). The sample of the tweets and the sentiment classes, according to Table 1 shows the sample Tweet Data for Sentiment

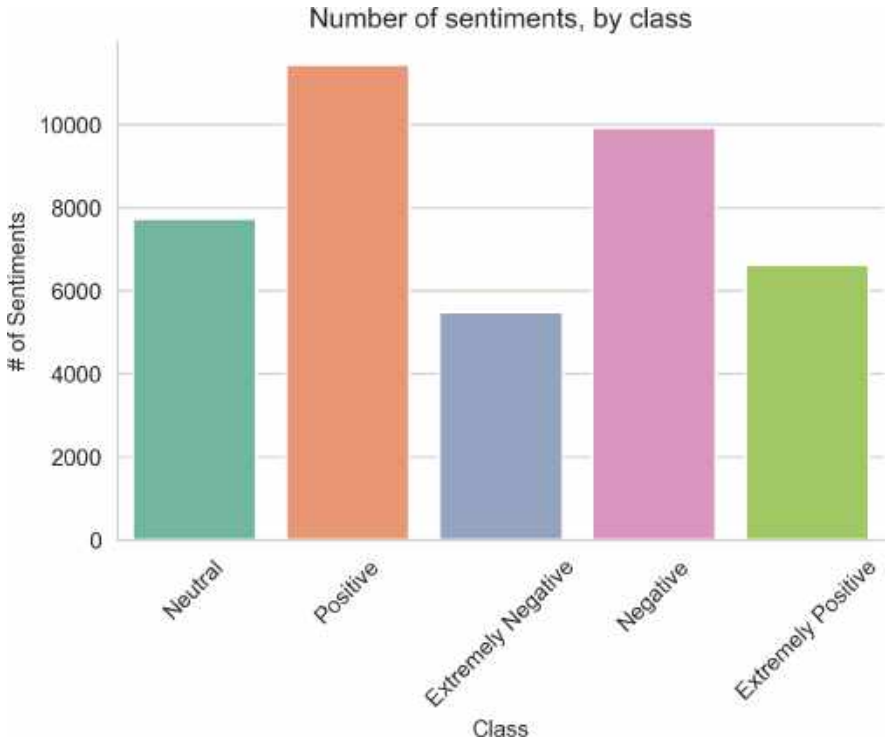


Fig. 2. Class distribution of the dataset

Classification, and Figure 2, shows the data distribution in each class for data with five classes.

1.3 Data Preprocessing

Raw data must be treated in a preprocessing stage before it can be successfully used with machine learning algorithms. This stage prepares the data to be used. This system performs its data preprocessing with the assistance of Natural Language Processing [24].

The text data are, first and foremost, changed to lowercase during this stage. This form has all stop words eliminated, and the corresponding contractions have been changed. In the Python NLTK package, a list of stop words is defined, which is used in this procedure.

Additionally, a custom function is developed to substitute contractions to finish the job. In order to reduce the likelihood of confusion, a check for spelling errors is carried out. The first step is to replace uppercase with lowercase. Following this step, the text will have any special characters, URLs, HTML tags, and stop words removed.

The text data is subjected to one more round of tokenization [14], normalization, and lemmatization. When it comes to natural language processing, there are three critical functions known as stemming, tokenization, and normalization used for preprocessing text before classification.

1. Tokenization: In natural language processing (NLP), tokenization divides text content into smaller components. A token is a name given to each unit. Every single word is turned into a token for this work [8].
2. Stemming: In stemming, the morphological forms of a word are converted back to their stems under the assumption that each form is semantically related to the others. The stem does not need to be a term already present in the dictionary. Nevertheless, after stemming is complete, all of the stem's variants should map

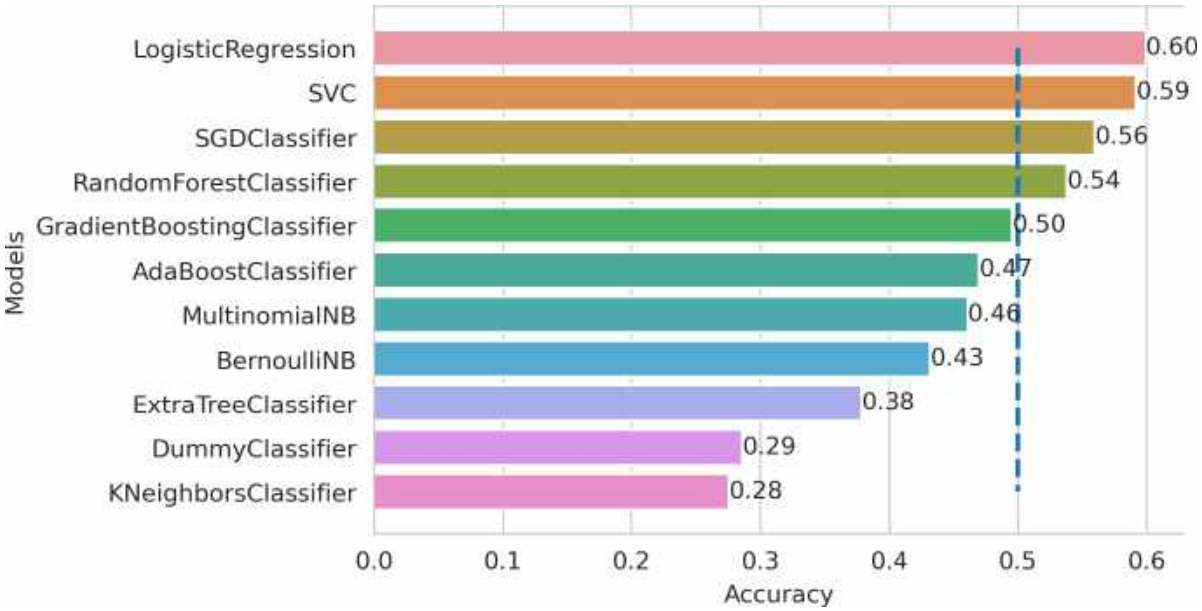


Fig. 3. The results obtained by the different machine learning models

to this form. When utilizing a stemmer, two things need to be taken into consideration [21]:

(a) It is reasonable to presume that the various morphological variants of a word have the same core meaning, and they should thus all be mapped to the same stem.

(b) It is essential not to confuse words that do not have the same meaning with one another.

These two rules are sufficient so long as the stems produced are helpful for the programs we use for text mining and language processing. In most contexts, stemming is understood to function as a mechanism that improves recall. Compared to languages with a more complex morphology, the influence of stemming is not as strong in languages with a relatively simple morphology.

3. Normalization: It is the process of converting an odd text into its typical form.

   People occasionally use a term unconventionally to convey their meaning [19]. This content has to be reformatted into its proper form, and any spelling errors need to be corrected.

4. Extracting Features: By extracting features from text and representing them as a vector of real numbers, a procedure known as “text feature extraction” can be performed [26, 10].

In this study, we used a technique called TF-IDF that generates a vector containing a set of real-valued features for each text, with the value of each feature depending on how often a specific word occurs in the text.

2 Building Models

Four different ML models were built using preprocessed tweets. The ML models were trained using the training dataset, while the performance of the models was evaluated using both the training and test datasets. The ML models are analyzed in detail in the following subsection.

Table 2. Performance measures for the proposed models

Models	Accuracy	Precision	Recall	F1-score
Voting Classifier	0.60	0.62	0.63	0.64
Bagging Classifier	0.61	0.61	0.63	0.65
Stacking Classifier	0.64	0.64	0.65	0.66
BERT (Base)	**0.73**	**0.74**	**0.74**	**0.74**

2.1 Analyzing Machine Learning Models

2.1.1 Voting

A voting ensemble technique is a machine learning model that produces a single final prediction by combining the predictions of multiple machine learning models [28]. Because all the training data were used to train the models with this ensemble method, they should each have their personality. When performing regression tasks, the result is the mean of the predictions made by the models.

Instead, two methods are available: hard voting and soft voting, which can be used to estimate the final output of classification problems. Voting's primary purpose is to enhance generality by correcting flaws specific to each model. This is especially important when the models perform well on a predictive modeling problem.

2.1.2 Bagging

Bagging subsamples of training data to improve one classifier's generalization performance. Overfitting models benefit from this strategy. Bagging data from subsamples includes bootstrapping and aggregating. This method uses random sampling with replacement to resample the data, which overlaps training data. Regression voting or classification voting yields the final prediction for each data set. This strategy improves very little because the classifier's hyperparameters do not vary from one subsample to another. This bias-reduction strategy is expensive and will not help with volatility. It reduces variance by better generalizing when the data is overfitted but not under fitted.

2.1.3 Stacking

Stacking ensemble models employ weighted voting to avoid all models contributing equally to the forecast. Stacking models have base models and meta-models (models that learn how to combine the predictions of the base models). Linear regression is used for regression, and logistic regression for classification. Out-of-sample base model predictions teach the meta-model. In other words, (1) data not used to train the base models are fed to them, (2) they make predictions, and (3) these predictions and the ground truth labels are utilized to fit the meta-model. Regression problems use predicted values. The affirmative class prediction is usually the input for binary classification problems. Finally, the multi-class classification uses the projected values for all classes.

2.1.4 BERT Classifier

BERT is a deep learning model that excels at NLP tasks. One output layer may fine-tune BERT's deep bidirectional representation [2]. This paper used BERT-Base. Moreover, BERT-Base has 12 layer/transformer blocks, 768 hidden units, and 12 self-attention heads with 110 M optimized parameters. BERT employs a 30000-word set of fundamental embeddings [30].

The input representation is the token, segment, and position embeddings total. Furthermore, for preprocessing data, both [CLS] and [SEP] were used as a classification token and a sentence marker, respectively. Additionally, the sentiment categorization output layer comprises [CLS] representation.

3 Experimental Results and Discussion

This section briefly discusses the study of different ML ensemble algorithms for the category of user sentiment under different labels (extremely positive, positive, extremely negative, negative, and neutral). The ML models were created and examined using the scikit-learn [25] package and the Python programming language.

The manually labeled dataset was split 80/20 randomly between the training and testing phases. As a result, 80% of the data were classified as training data, and 20% as testing data. The grid search tuning approach [7] was used to tune the hyperparameters, which can regulate how the algorithms learn, to identify the best hyperparameters for the utilized models.

The algorithms' performance was evaluated using precision, recall, and the F1-score. The experiment was conducted to discover the best parameters for each method used to classify the sentiment data of the COVID-19 tweets. Tweets with five classes were used in the experiment.

The first time we used the popular machine learning algorithms with the data representation of TF-IDF, the figure 3 shows the results obtained, such as the algorithms used: Logistic Regression (Lr), Support Vector Machine (SVM), Naïve Bayes (NB), K-Nearest Neighbors (KNN), Random Forest, Gradient Boosting, and AdaBoosting. Then the best models (such as those with accuracy above 50%) were selected to build the ensemble models (voting, bagging, and staking). Finally, the BERT models were used.

3.1 Voting Classifier (VC) Setup

To obtain the final predicted labels, hard voting, also known as majority voting, was used in this study among the Decision Tree (DT), Support Vector Classifier (SVC), and Logistic Regression (LR). The precision, recall, and F1-score for the VC model on the test dataset were 98.9%, 99.5%, and 99.3%, respectively.

3.2 Bagging Classifier (BC) Setup

The outputs from the predictive models are then applied to a voting scheme for better categorization. The basic estimator for training the BC model in this investigation was a Logistic Regression with $n_{\text{estimators}} = 100$. The bagging classifier's accuracy, precision, recall, and F1-score on the testing dataset were 61%, 61%, 63%, and 61%, respectively (see Table 2).

3.3 Stacking Classifier (SC) Setup

The proposed SC model's design consisted of two levels. The VC and BC models discussed above made up the first layer of the SC model, and a logistic regression model made up the second layer.

For every observation and test in the dataset, two distinct models were used to generate the conclusions. The judgments attained by these methods served as input features for the second-layer LR model.

The second-layer model then delivered the result based on the input features. The SC model's accuracy, precision, recall, and F1-score were 64%, 64%, 65%, and 65% on the training set, respectively (see Table 2).

3.4 BERT Setup

The BERT process was divided into two stages: pre-training and fine-tuning. The BERT architecture was trained on several tasks using unlabeled data during the pre-training phase. This was achieved so that it could be used later. Then, for fine-tuning, BERT was trained on the data utilized in this research, namely the tweets from the COVID-19 event. During the fine-tuning process, the used parameters included learning rates of 10-5, a batch size of 32, and a maximum iteration of 15 epochs.

Within the framework of the sentiment categorization method that uses BERT, the following parameters were observed: (a) the total number of effective classes is five; (b) the learning rate is 10e-5. Table 2 and figure 4 present the findings of the performance evaluation of the sentiment categorization using BERT.

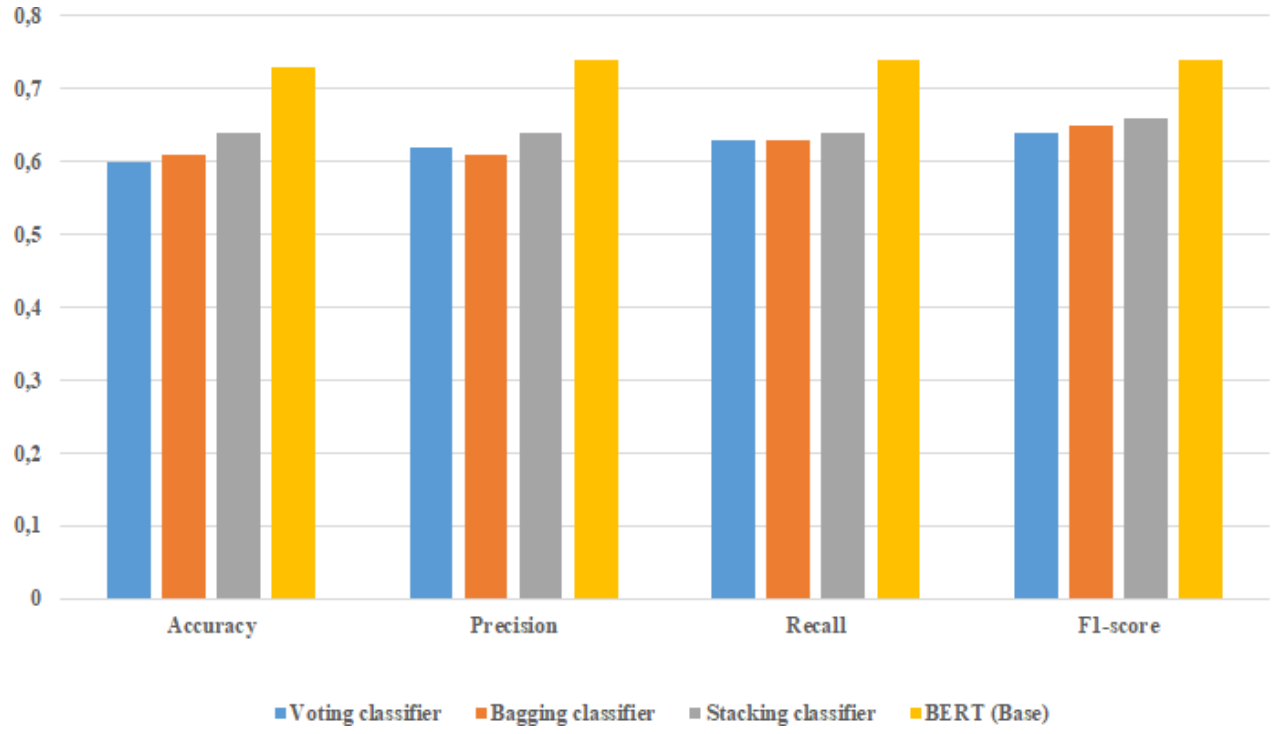


Fig. 4. Performance of different models

The best performance achieved was 0.74 (precision), 0.74 (recall), and 0.74 (F1-score) for classifying the five sentiment classes.

4 Conclusion

In this paper, the sentiment classification of the COVID-19 tweets dataset was investigated by comparing two sentiment classification schemes. The first scheme included ensemble ML models to classify tweets into five classes.

The Stacking Classifier showed the highest F1 score of 65% in this scheme, while Voting Classifier and Bagging Classifier models showed promising results, indicating that ensemble ML models can be used for sentiment analysis. The second scheme is sentiment classification using BERT.

The classification results achieved by BERT were better than the first scheme, reaching 74% (F1-score), 74% (precision), and 74% (recall) for the classification of five sentiment classes. Future studies may focus on trying different encoders, such as the variants of BERT and Word2vec, for text embedding to find the best suitable encoding for the classifiers and get better outcomes.

References

1. **Abd-Alrazaq, A., Alhuwail, D., Househ, M., Hamdi, M., Shah, Z. (2020).** Top concerns of tweeters during the COVID-19 pandemic: Infoveillance study. Journal of Medical Internet Research, Vol. 22, No. 4, pp. e19016. DOI: 10.2196/19016.
2. **Bozuyla, M., Ozundefinedift, A. (2022).** Developing a fake news identification model with advanced deep languagetransformers for turkish covid-19 misinformation data. Turkish Journal of Electrical Engineering and Computer Sciences, Vol. 30, No. 3, pp. 908–926. DOI: 10.55730/1300-0632.3818.
3. **Chakraborty, K., Bhatia, S., Bhattacharyya, S., Platos, J., Bag, R., Hassanien, A. E.**

(2020). Sentiment analysis of covid-19 tweets by deep learning classifiers—a study to show how popularity is affecting accuracy in social media. Applied Soft Computing, Vol. 97, pp. 106754. DOI: 10.1016/j.asoc.2020.106754.

4. **Chong, W. Y., Selvaretnam, B., Soon, L. K. (2014).** Natural language processing for sentiment analysis: An exploratory analysis on tweets. 4th International Conference on Artificial Intelligence with Applications in Engineering and Technology, pp. 212–217. DOI: 10.1109/icaiet.2014.43.
5. **Dhaya, R. (2020).** Deep net model for detection of COVID-19 using radiographs based on ROC analysis. Journal of Innovative Image Processing, Vol. 2, No. 3, pp. 135–140. DOI: 10.36548/jiip.2020.3.003.
6. **Garcia, K., Berton, L. (2021).** Topic detection and sentiment analysis in twitter content related to covid-19 from Brazil and the USA. Applied Soft Computing, Vol. 101, pp. 107057. DOI: 10.1016/j.asoc.2020.107057.
7. **Ghawi, R., Pfeffer, J. (2019).** Efficient hyperparameter tuning with grid search for text categorization using kNN approach with BM25 similarity. Open Computer Science, Vol. 9, No. 1, pp. 160–180. DOI: 10.1515/comp-2019-0011.
8. **Ghulam, H., Zeng, F., Li, W., Xiao, Y. (2019).** Deep learning-based sentiment analysis for roman urdu text. Vol. 147, pp. 131–135. DOI: 10.1016/j.procs.2019.01.202.
9. **Gupta, M., Bansal, A., Jain, B., Rochelle, J., Oak, A., Jalali, M. S. (2021).** Whether the weather will help us weather the covid-19 pandemic: Using machine learning to measure twitter users' perceptions. International Journal of Medical Informatics, Vol. 145. DOI: 10.1016/j.ijmedinf.2020.104340.
10. **Imran, M., Afzal, M. T., Qadir, M. A. (2017).** A comparison of feature extraction techniques for malware analysis. Turkish Journal of Electrical Engineering and Computer Sciences, Vol. 25, pp. 1173–1183. DOI: 10.3906/elk-1601-189.
11. **Islam, M. N., Inan, T. T., Rafi, S., Akter, S. S., Sarker, I. H., Islam, A. K. M. N. (2020).** A systematic review on the use of AI and ML for fighting the COVID-19 pandemic. IEEE Transactions on Artificial Intelligence, Vol. 1, No. 3, pp. 258–270. DOI: 10.1109/tai.2021.3062771.
12. **Islam, M. N., Mahmud, T., Khan, N. I., Mustafina, S. N., Najmul-Islam, A. K. M. (2021).** Exploring machine learning algorithms to find the best features for predicting modes of childbirth. IEEE Access, Vol. 9, pp. 1680–1692. DOI: 10.1109/ACCESS.2020.3045469.
13. **Islam, M. N., Najmul-Islam, A. K. M. (2020).** A systematic review of the digital interventions for fighting COVID-19: The Bangladesh perspective. IEEE Access, Vol. 8, pp. 114078–114087. DOI: 10.1109/access.2020.3002445.
14. **Javed-Mehedi-Shamrat, F. M., Tasnim, Z., Ghosh, P., Majumder, A., Hasan, M. Z. (2020).** Personalization of job circular announcement to applicants using decision tree classification algorithm. IEEE International Conference for Innovation in Technology, pp. 1–5. DOI: 10.1109/inocon50539.2020.9298253.
15. **Kaur, H., Ahsaan, S. U., Alankar, B., Chang, V. (2021).** A proposed sentiment analysis deep learning algorithm for analyzing covid-19 tweets. Information Systems Frontiers, Vol. 23, No. 6, pp. 1417–1429. DOI: 10.1007/s10796-021-10135-7.
16. **Kazi-Shahrukh, O., Prodipta, M., Nabila-Shahnaz, K., Md.-Rezaul, K. R., Md-Nazrul, I. (2019).** A machine learning approach to predict autism spectrum disorder. pp. 1–6. DOI: 10.1109/ECACE.2019.8679454.
17. **Khan, R., Shrivastava, P., Kapoor, A., Tiwari, A., Mittal, A. (2020).** Social media analysis with AI: Sentiment analysis

techniques for the analysis of twitter COVID-19 data. Journal of Critical Reviews, Vol. 7.

18. **Laato, S., Islam-Najmul, A. K. M., Islam, M. N., Whelan, E. (2020).** What drives unverified information sharing and cyberchondria during the COVID-19 pandemic?. European Journal of Information Systems, Vol. 29, No. 3, pp. 288–305. DOI: 10.1080/0960085x.2020.1770632.
19. **Mehta, R. P., Sanghvi, M. A., Shah, D. K., Singh, A. (2019).** Sentiment analysis of tweets using supervised learning algorithms. Springer Singapore, pp. 323–338. DOI: 10.1007/978-981-15-0029-9_26.
20. **Miglani, A. (2020).** Coronavirus tweets NLP - Text classification. www.kaggle.com/datasets/datatattle/covid-19-nlp-text-classification.
21. **Moral, C., de-Antonio, A., Imbert, R., Ramírez, J. (2014).** A survey of stemming algorithms in information retrieval. Information Research, Vol. 19, No. 1.
22. **Nemes, L., Kiss, A. (2020).** Social media sentiment analysis based on covid-19. Journal of Information and Telecommunication, Vol. 5, No. 1, pp. 1–15. DOI: 10.1080/24751839.2020.1790793.
23. **Nichols, J. A., Herbert-Chan, H. W., Baker, M. A. B. (2019).** Machine learning: applications of artificial intelligence to imaging and diagnosis. DOI: 10.1007/s12551-018-0449-9.
24. **Pang, B., Lee, L. (2004).** A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. pp. 271–278. DOI: 10.3115/1218955.1218990.
25. **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Müller, A., Nothman, J., Louppe, G., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E. (2012).** Scikit-learn: Machine learning in python. Journal of Machine Learning Research, Vol. 12, pp. 2825–2830.
26. **Perkins, J. (2014).** Python 3 text processing with NLTK 3 cookbook. Packt Pub Ltd.
27. **Rustam, F., Khalid, M., Aslam, W., Rupapara, V., Mehmood, A., Choi, G. S. (2021).** A performance comparison of supervised machine learning models for covid-19 tweets sentiment analysis. PLOS ONE, Vol. 16, No. 2, pp. e0245909. DOI: 10.1371/journal.pone.0245909.
28. **Ruta, D., Gabrys, B. (2005).** Classifier selection for majority voting. Information Fusion, Vol. 6, No. 1, pp. 63–81. DOI: 10.1016/j.inffus.2004.04.008.
29. **Villavicencio, C., Macrohon, J. J., Inbaraj, X. A., Jeng, J. H., Hsieh, J. G. (2021).** Twitter sentiment analysis towards COVID-19 vaccines in the Philippines using Naïve Bayes. Information, Vol. 12, No. 5, pp. 204. DOI: 10.3390/info12050204.
30. **Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., et al. (2016).** Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv, pp. 1–23. DOI: 10.48550/arXiv.1609.08144.
31. **Zaman, A., Muhammad-Nazrul, I., Zaki, T., Sajjad-Hossain, M. (2020).** ICT intervention in the containment of the pandemic spread of COVID-19: An exploratory study. arXiv, pp. 1–16. DOI: 10.48550/arXiv.2004.09888.

Article received on 08/04/2023; accepted on 18/04/2024.
** Corresponding author is Maaskri Moustafa.*

ISSN 2007-9737

Comparative Study for Text Chunking Using Deep Learning: Case of Modern Standard Arabic

Nabil Khoufi[1,*], Chafik Aloulou[2]

[1] University of Sfax, ANLP Research Group, Sfax, Tunisia

[2] University of Sfax, MIRACL Lab, Sfax, Tunisia

nabil.khoufi@outlook.com, chafik.aloulou@fsegs.usf.tne

Abstract. The task of chunking involves dividing a sentence into smaller phrases by identifying a limited amount of syntactic information. This process involves grouping together consecutive words to form phrases, also known as shallow parsing. Chunking does not provide information on the relationships between these phrases. This paper describes our approach to building chunking models for Arabic text using deep learning techniques. We evaluated several training models and compared their results using a rich data set. The results we obtained were highly encouraging when compared to previous related studies.

Keywords. NLP, Arabic language, shallow parsing, chunking, deep learning, GRU, LSTM, BILSTM, ATB, Penn Arabic treebank.

1 Introduction

Text parsing is a critical aspect of natural language processing (NLP) and has received significant attention since the early days of NLP. The information generated by parsing is valuable for various NLP applications, such as automatic summarization, author profiling and named entity recognition [1]. Parsing can either be shallow or deep. Shallow parsing, also known as chunking, focuses on identifying the boundaries of larger constituents or phrases, while deep parsing goes further by identifying both the constituents and their internal structure. The two types of parsing require different amounts of information and produce different results.

The chunking task can be approached through two main methods: the grammar-based approach and the machine learning approach [2] The former uses a set of grammatical rules, while the latter employs machine learning techniques and relies on annotated data. This paper details our experiments on chunking Arabic text using various deep learning architectures. The structure of this paper is as follows:

In Section 2, we outline the fundamental concepts of the chunking task. Section 3 discusses the syntactic ambiguities in Arabic. Section 4 reviews prior research on chunking in Arabic. In Section 5, we describe our deep learning models and approach to chunking Arabic text. Section 6 presents the evaluation process and results obtained. Finally, in Section 7, we present our conclusions and suggest avenues for future research.

2 Chunking Task Background

In 1991, Steven Abney proposed an approach to parsing that involves identifying groups of words that are syntactically correlated [3]. He argued that when we read, we do so in chunks, and therefore chunking involves dividing a sentence into smaller parts that are syntactically related. Chunking can be seen as an intermediate step towards full parsing as it provides part of the complete syntactic structure of a sentence.

Initially, the chunking task focused on recognizing noun phrases (NPs), which is known as noun phrase chunking. Lance Ramshaw and Mitch Marcus tackled NP-chunking using a

machine learning method [4]. They recognized various chunks but categorized every chunk that was not a NP as a VP chunk.

This work inspired many other studies that have investigated the application of learning methods to noun phrase chunking. Later, researchers focused on other constituents of sentences such as VP, PP, ADJP, or ADVP to provide a more comprehensive description of the sentence. The following example shows a chunked Arabic sentence [5].

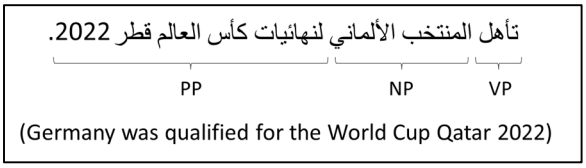


Fig. 1. An instance of chunking in modern standard Arabic

3 Sources of Ambiguity in Arabic Language

Arabic, like all Semitic languages, has a complex morphology and a vast vocabulary, making it more challenging to parse than other natural languages [6]. In addition to common linguistic features like coordination, anaphora, and ellipsis found in Latin-based languages, Arabic has unique characteristics that pose difficulties in the parsing process [7].

3.1 Unvocalisation

The lack of vowels in written words, known as unvocalization, leads to grammatical ambiguity. Words without vowels in their written representation can't effectively differentiate between different grammatical interpretations and meanings, as a single word can have multiple grammatical variations.

As a result, unvocalized text is more ambiguous than text with vowels [8, 9]. According to Debili's statistics (Debili et al., 2002), 74% of Arabic words have more than one vocalization option.

The ambiguity rate for grammatical interpretation is higher in unvocalized words, with an average rate of 8.7, compared to an average rate of 5.6 for vocalized words. Table 1 provides an example of a single unvocalized word with its various vocalized forms.

3.2 Agglutination

In Arabic, there is a distinct occurrence known as agglutination, where words such as articles, prepositions, pronouns, etc. can be attached to adjectives, nouns, verbs, and particles that they are associated with.

This leads to complex syntax, resulting in unusual sentence structures. In fact, an agglutinative form can even make up an entire sentence, as demonstrated in table 2. In such instances, specific processing is necessary to determine the correct syntactic structure of the sentence.

3.3 Words Order

The arrangement of words in Arabic is flexible. Typically, the word that is intended to be emphasized is placed at the beginning of the sentence and the word with the most meaning or tone is placed at the end. This freedom in word order results in artificial syntactic confusion and makes constructing grammar more difficult.

To account for all possible correct word arrangements in a sentence, grammar rules must include all combinations. Table 3 demonstrates the impact of changing the order of words. The order of words in this sentence can be rearranged, resulting in the two structures shown in Table 4 and 5.

3.4 Recursive Structure

The frequent use of recursive structures is another characteristic of Arabic texts. The presence of nested structures is common in Arabic as well as in other natural languages, but it occurs more frequently in Arabic due to some propositions being able to play a role within other propositions. For example:

الشرطة هي التي قبضت على المجرم الذي ضل هارباً مدة طويلة.

(The police have arrested the criminal who remained on the run for a long time).

Table 1**.** An illustration of ambiguity due to the unvocalization phenomenon

Unvocalized Word	Vocalized Forms	Buckwalter Transliteration	Translation
فهم	فَهِمَ	fahima	He understood
	فَهَّمَ	fah~ama	He explained
	فُهِمَ	fuhima	It has been understood
	فَهْمٌ	fahomN	Comprehension
	فَهُمْ	fahumo	Then them
	فَهَمَّ	faham~a	Then started
	…	…	…

Table 2. A sample of a sentence in an agglutinative form (one word)

Sentence	Buckwalter transliteration	Gloss
واستقبلهم	wastaqbalahum	(Then he welcomed them).

Table 3. Arabic sentence, order 1

Arabic sentence	إلى بـعلملا	الولد	ذهب
English translation	to the stadium	the boy	went
Form	complement	subject	verb

Table 4. Arabic sentence, order 2

Arabic sentence	إلى بـعلملا	ذهب	الولد
English translation	to the stadium	went	the boy
Form	complement	verb	subject

Table 5. Arabic sentence, order 3

Arabic sentence	الولد.	ذهب	إلى بـعلملا
English translation	the boy	went	to the stadium
Form	subject	verb	complement

It is a nominal sentence, while the proposition (خبر) is also a nominal sentence:

هي التي قبضت على المجرم الذي ضل هارباً مدة طويلة

(Have arrested the criminal who remained on the run for a long time).

In the aforementioned example, it is even challenging to segment the text into sentences because of the numerous propositions that are interdependent and do not belong to the same syntactic level. This lack of independence leads to Arabic sentences being of unlimited length.

4 Related Works

Compared to the research conducted in English and other languages, there is a scarcity of studies on the Arabic chunking problem. Only four works can be found that specifically address the Arabic chunking task. In 2004, Mona Diab and colleagues [11, 12] carried out tokenization, POS tagging, and used an SVM-based method for Arabic text chunking.

They utilized an existing SVM tool [13]. The features used in their system were words and POS annotations, along with a context window of -2/+2. The system was evaluated on 400 sentences and produced a chunking performance of 92.06% precision, 92.09% recall, and 92.08 F-measure. This research was the first of its kind. Diab later used the same SVM tool and trained their model using the Arabic Treebank with a modified POS tag set [12].

They reported an F-measure chunking performance of 96.33%. Mohammed and Omar

Table 6. Comparative summary of related works

Works	Approach	Training Data	Testing data	Results
[12]	Machine learning-Based	18 970 sentences	2337 sentences	F-measure 91.44%
[14]	Grammar-Based	-	70 sentences	F-measure 97%
[15]	Machine learning-Based	2300 words	283 sentences	Accuracy 80.46%
[16]	Machine learning-Based	10 100 sentences	2524 sentences	Accuracy 96.54% F-measure 76,23% Recall 73,86% Precision 81,57%

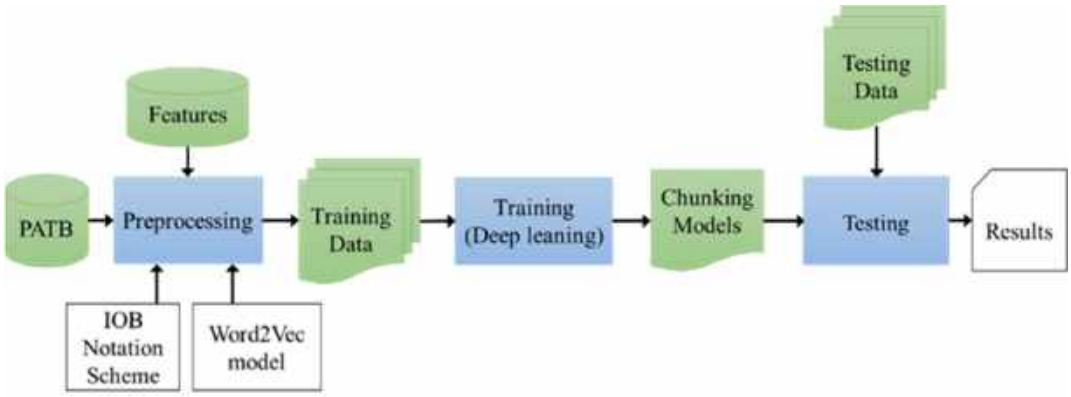


Fig. 2. Proposed method architecture

Table 7. Features list

Feature	Description
w[t]	the word being proceeded
w[t+1]	the word on right vicinity at position t+1
w[t+2]	the word on right vicinity at position t+2
pos[t]	the POS annotation of w[t]
pos[t+1]	the POS annotation of w[t+1]
pos[t+2]	the POS annotation of w[t+2]

[14] describe the development of an Arabic shallow parser based on a rule-based approach.

The chunking which constitutes the main contribution are achieved on two successive stages that include grouped sequences of adjacent words based on linguistic properties to identify each of NPs, VPs and PPs. Since the aim of the research is to generate results at two levels, the final results adopted were based on the second level results.

Tested on only 70 sentences, their system achieved F-measure of 97%. Ben-Fraj and Kessentini [15] proposed an approach for chunking Arabic texts based on a combinatorial classification process.

It is a modular chunker that identifies the chunk heads using a combinatorial binary classification before recognizing their types based on the parts-of-speech (POS) of the chunk heads, already identified.

For the experimentation, the authors used 226 sentences as training data. They obtained 80.46% accuracy for the full chunking process.

Table 8. An instance of an annotated sentence based on the IOB notation model

Sentence	.	2016	العربية	الثقافة	عاصمة	صفاقس
Transliteration	.	2016	AlErbyp	AlvqAfp	EASmp	SfAqs
Traduction	.	2016	Arab	Culture	Capital	Sfax
Annotation IOB	O	I-NP	I-NP	I-NP	B-NP	B-NP

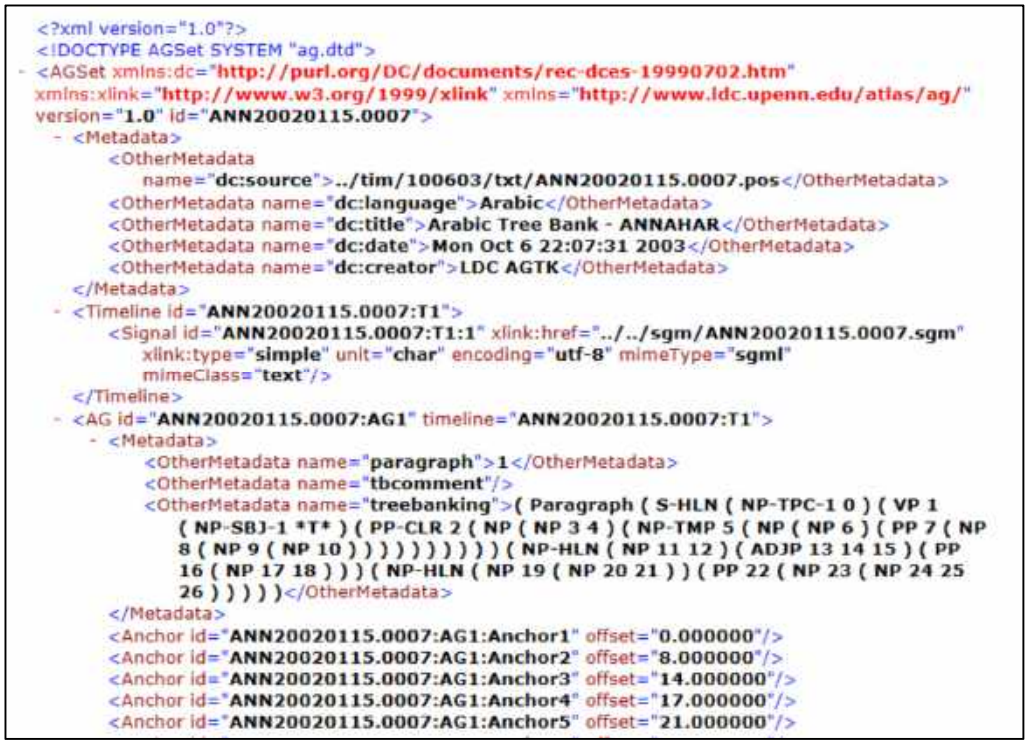


Fig. 3. A segment of the training corpus in its original XML tree format

Khoufi et al. developed a machine-learning based model for Arabic text chunking [16]. They utilized CRFs (Conditional Random Fields) for training the model.

The training data was derived from the PATB syntactic trees (80% of the ATB) and included words and their part of speech annotations as features, with a context window of -2/+2 words centered around the word being processed.

The model also considered bigrams and trigrams of words and POS annotations.

The model was tested on 20% of the PATB corpus, representing 2524 sentences, and achieved an accuracy of 96.54%.

A summary of their study of Arabic chunkers is presented in Table 6.

5 Proposed Method

This section presents the design of our proposed method. Our approach is based on deep learning technology for chunking Arabic text, as opposed to a rule-based method.

This is because constructing a grammar that encompasses all the exceptional and specific syntactic structures in Arabic is highly challenging, if not impossible. Our method involves using an annotated corpus, a set of features, and a model, which will be discussed in the subsequent section.

Fig. 4. Evaluation results comparison

Fig. 5. Accuracy results

Figure 2 illustrates the design of our approach, where a portion of the annotated corpus (80%) undergoes pre-processing using features, the Word2Vec model, and the IOB notation scheme.

This generates a model that is used to analyze sentences. The model's performance is then evaluated using the remaining portion of the annotated corpus (20%) through cross-validation. Detailed information regarding our method can be found in subsequent sub-sections.

5.1 Our Features

The selection of features is crucial in machine-learning algorithms as it has a significant impact on the accuracy of labeling. Features determine what information is extracted from the annotated corpus during the training phase. In this study, we chose to use the word itself (W) and its Part-of-Speech (POS) annotation as our features.

These features capture the characteristics of the word at position t by utilizing information from the surrounding words. The features utilized for the training phase are shown in Table 7.

5.2 Used Tag Set

For our experiment, we employ the tag set of the PATB, which consists of 23 tags: S, NP, VP, SQ, PP, SBAR, SBARQ, NX, PRN, PRT, QP, ADJP, ADVP, FRAG, WHNP, WHPP, WHADJP, WHADVP, CONJP, INTJ, NAC, UCP, X.

In addition to these tags, we use the IOB notation model, where each word is tagged with a chunk label and one of three additional tags:

- B for the first word of a chunk,
- I for a non-initial word in a chunk,
- and O for a word outside of any chunk.

This increases the number of tags in the tag set to 46, as each tag in the tag set becomes either an I or B tag. For example, NP can be represented as two chunk types, I-NP or B-NP. Table 8 provides an example of an Arabic sentence tagged using the IOB scheme, with the English Translation shown in the right-to-left direction.

5.3 Training Corpus

The Penn Arabic Treebank was established by the Linguistic Data Consortium (LDC) at the University of Pennsylvania [17]. It is composed of data obtained from standard and modern Arabic linguistic sources, consisting of 402,291 tokens and 12,624 sentences.

The texts in the corpus do not contain any vowels, as is typical in most written Arabic texts. In our experiments, we used version 3.2 of this corpus. In order to perform the training phase, the training corpus must be pre-processed to incorporate the IOB notation scheme and the selected features along with their context window.

This pre-processing step transforms the PATB corpus from its original tree format into a vector format. Figure 3 displays the original format of the training corpus.

5.4 Training Experiments

For the training stage, we utilize the Word2Vec model for constructing word embeddings. Word2Vec is a widely used method for building word embeddings and was first introduced by [18]. There are two variations of the Word2Vec model for learning word embeddings:

The Skip-gram and CBOW (Continuous Bag of Words) models. Each of these models consists of three layers: an input layer, a hidden layer, and an

Fig. 6. Results comparation between our DL models and CRF model

output layer, with the output layer consisting of neurons with a SoftMax activation function.

- The CBOW architecture enables prediction of a word based on its context, using a word window to the left and right. This model assumes that the order of the context words has no effect on the projection, and the projection layer is shared among all words. Learning word embeddings with the CBOW architecture involves predicting a word based on its context, by calculating the vector obtained by summing the embeddings of the context words, and then applying a log-linear classifier to predict the target word.

- The Skip-gram architecture is also a three-layer log-linear neural network. Unlike the CBOW model, it allows for predicting a context window given the word at the center of the context. The central word serves as the input to the network, and the words in the context form the output. The goal is to predict, for a given word, its context, so that the embedding of any word is close to the embeddings of words in the same context.

In this study, we utilized the CBOW architecture for word embedding construction as it is a widely used method in NLP and has produced positive outcomes in previous NLP studies such as [19]. Today, Recurrent Neural Networks (RNNs) are the most commonly used systems in various machine learning tasks.

They are frequently utilized in computer vision (such as image classification, object detection, segmentation, etc.) and natural language processing (such as automatic translation, voice recognition, language models, etc.). In our Arabic chunking task, we experiment with three different RNN architectures (LSTM, BILSTM, GRU), which we present below, to compare their performance.

5.4.1 LSTM

Long Short-Term Memory (LSTM) is a deep learning-based RNN architecture that has feedback connections, unlike regular feedforward neural networks. LSTM can handle not just individual data points (such as images), but also sequential data (such as speech or video) [20].

LSTMs are designed to address the vanishing gradient problem that occurs in traditional RNNs, which can make it difficult for the network to capture long-term dependencies in sequential data. In an LSTM, the network has a hidden state that is updated at each time step.

The update is controlled by three gates: the input gate, the forget gate, and the output gate. The input gate determines how much of the new input should be added to the current hidden state, the forget gate determines how much of the previous hidden state should be forgotten, and the output gate determines how much of the new hidden state should be output.

Applications of LSTM include unsegmented handwriting recognition [21], speech recognition [22], and network traffic anomaly detection or intrusion detection systems (IDSs).

5.4.2 BILSTM

Bidirectional Long Short-Term Memory (BILSTM) is a type of recurrent neural network architecture. In a traditional LSTM, the input sequence is processed in one direction, from the beginning to the end. However, in a BILSTM, the input sequence is processed in two directions simultaneously: one forward and one backward.

The BILSTM consists of two LSTM layers, one processing the input sequence in the forward direction and the other in the backward direction. The outputs of both the layers are concatenated at each time step to form the final output of the BILSTM.

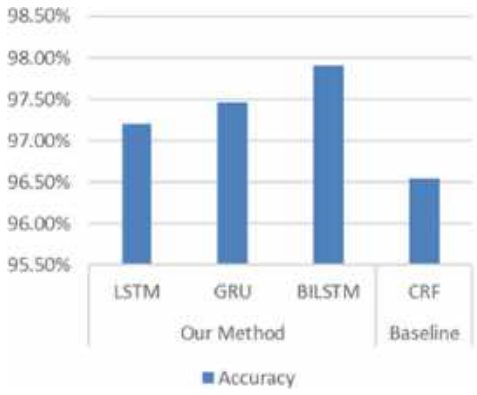


Fig. 7. Accuracy comparison between our DL models and our baseline

This allows the model to capture both the past and the future context of each input token, which can be useful in many sequence modelling tasks. The BILSTM has been widely used in various natural language processing tasks such as part-of-speech tagging, named entity recognition, sentiment analysis, and machine translation [23], among others. It has been shown to outperform traditional LSTM models in many of these tasks.

5.4.3 GRU

Gated Recurrent Units (GRUs) are gating mechanisms in RNNs, first introduced in 2014 by Kyunghyun Cho et al. [24]. GRUs are similar to LSTMs, but have a simpler structure with fewer parameters, making them faster to train and less prone to overfitting [25]. In a GRU, the network has a hidden state that is updated at each time step.

The update is controlled by two gates: the reset gate and the update gate.

The reset gate determines how much of the previous hidden state should be forgotten, while the update gate determines how much of the new input should be added to the current hidden state.

The update gate and reset gate are both sigmoid functions that take as input the current input and the previous hidden state.

The output of these gates is then used to update the hidden state. Unlike LSTMs, which have separate memory cells, GRUs use a single hidden state to store information about the input sequence.

GRUs have shown comparable performance to LSTMs in tasks such as polyphonic music modelling, speech signal modelling, and natural language processing [26], and even better performance on smaller and less frequent datasets [27].

6 Evaluation Results and Discussion

In order to evaluate the models, we divided the PATB corpus into two parts, with 80% consisting of 10,100 sentences for training and 20% consisting of 2,524 sentences for testing.

The unvocalized version of the Treebank was used for all experiments and all the data was sourced from the parsed trees in the Treebank.

By using the unvocalized version of the Treebank, we ensure that our results are consistent and comparable with other studies that use the same corpus.

We measured the performance of the model's using precision, recall, F-measure and accuracy.

These metrics allows us to have a comprehensive view of the performance of the models and help us in choosing the best model for our Arabic chunking task. This evaluation method allows us to assess the ability of the models to correctly predict the chunk tags for the sentence chunks.

The results of these metrics are displayed in figure 4. When comparing the performance of the LSTM, GRU, and BILSTM models based on their precision, recall, and F-measure metrics, the results show that the BILSTM model achieved the highest precision (86.71%), recall (84.54%), and F-measure (85.59%).

This represents an enhancement of 6,37%, 5,94% and 6,13% respectively over the LSTM model's precision (80.34%), recall (78.60%), and F-measure (79.46%).

Similarly, the GRU model achieved an enhancement of 0.43%, 0.73%, and 0.63% respectively over the LSTM model's precision, recall, and F-measure by reaching a precision of 80,69%, a recall of 79,33% and an F-measure of 80%.

In addition, it's important to note that the BILSTM model achieved the highest performance in all three metrics, suggesting that it may be the most effective model overall.

Table 9. Chunking performance of BILSTM model calculated on major chunking tags

Tags	Precision	Recall	F-measure
NP	93,32%	95,55%	94,42%
VP	93,82%	94,11%	93,96%
PP	65,80%	65,94%	65,87%
ADJP	91,25%	86,83%	88,99%
ADVP	55,84%	57,41%	56,61%
CONJP	98,87%	98,83%	98,85%
PRT	95,82%	97,47%	96,64%
PRN	52,22%	48,38%	50,23%
O	96,67%	97,97%	97,32%
S	92,88%	96,87%	94,83%

This could be explained by the ability of BILSTM model to capture both past and future context when processing sequential data. BILSTM model results are confirmed by the accuracy metric.

Indeed, BILSTM model reached an accuracy of 97.90% slightly exceeding LSTM and GRU models, respectively 97.21% and 97.46%. Accuracy values comparison are displayed in the following figure 5. It also interesting to compare obtained results with previous work which we consider as our baseline.

We have developed a machine learning model for chunking Arabic using Conditional Random Fields (CRF) (Khoufi et al. 2015). We used the same data for training and testing the CRF model. Results comparisons are illustrated in the following Figure 6.

As shown in Figure 5, the LSTM, GRU, and BILSTM models outperformed the CRF model, which achieved an accuracy of 81.57%, a recall of 73.86%, and an F-measure of 76.23%. Indeed, among the DL-based models, BILSTM achieved the best results and significantly improved our baseline with a 5.14% increase in precision, a 10.68% increase in recall, and a 9.36% increase in F-measure.

Even the accuracy measurement confirms the superiority of the BILSTM model (97,90%) over the CRF model (96,54%), as demonstrated in Figure 7 below. For a detailed idea about the performance of our model, we calculated the precision, the recall and the f-measure for the major chunking tags.

These results are exposed in the following Table 9. In relation to the outcomes displayed in Table 9, our model has successfully identified significant portions with acceptable accuracy.

We want to emphasize that the model has achieved a commendable overall performance in identifying chunks, which validates the obtained results. As shown in Table 9, the CONJP category is recognized with the highest precision of 98.87%, recall of 99.47%, and f-measure of 98.85%.

This is reasonable due to the limited occurrence of conjunctions in Arabic. The model also performs well in recognizing PRTs, with a precision of 95.82%, recall of 97.47%, and f-measure of 96.64%. PRTs are typically associated with CONJP, which facilitates their identification.

However, we have observed some difficulties in detecting PP and ADVP chunks, with f-measures of 65.87% and 56.61%, respectively. PPs consist of a preposition followed by an object of preposition, such as NP, and this relationship with other chunks may make it challenging to determine their boundaries.

We attempted to compare our findings with those of other studies, but encountered difficulties in performing an accurate analysis because they used different metrics and datasets.

However, our accuracy value was higher than that of (Ben Fraj et al. 2012), with a 17% increase.

Also, the authors used only 2300 words as training data which is not sufficient to obtain a viable model.

We found that the testing data used by (Mohamed et al. 2011) was insufficient to provide an accurate assessment of system performance, as they only tested on 70 sentences compared to our model's 2,524 sentences. Despite this, (Diab et al. 2007) achieved a higher F-measure of 96.33% using a testing set of 2337 sentences compared to our F-Measure.

In summary of this study's results, it can be said that models based on deep learning techniques achieve good results in processing the chunking task of Arabic texts. These models, especially the BILSTM model, improved our results compared to the classical models that used traditional machine learning algorithms, such as the CRF model we used for our baseline.

7 Conclusion

In this study, we presented our approach for chunking Arabic texts through the use of deep learning models. The models we built are LSTM, BILSTM, and GRU, and they are constructed with morphosyntactic features and the IOB notation system.

The training data for the models was obtained from the PATB corpus. Our models were trained on 80% of the PATB and tested on the remaining 20%. Evaluation results shows the supremacy of BILSTM model for this task with an f-measure of 85.59% and an accuracy of 97.90%.

References

1. **Khoufi, N. (2017).** Une approche hybride pour l'analyse syntaxique de la langue arabe. Doctoral dissertation, Université de Sfax Tunisie.
2. **Khoufi, N., Aloulou, C., Belguith, L. H. (2016).** Toward hybrid method for parsing modern standard Arabic. 2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD, IEEE, pp. 451–456. DOI: 10.1109/SNPD. 2016.7515939.
3. **Abney, S. P. (1991).** Parsing by chunks. Principle-based parsing, Springer, Dordrecht, Vol. 44, pp. 257–278. DOI: 10.1007/978-94-011-3474-3_10.
4. **Ramshaw, L. A., Marcus, M. P. (1999).** Text chunking using transformation-based learning. In: Armstrong, S., Church, K., Isabelle, P., Manzi, S., Tzoukermann, E., Yarowsky, D. (eds) Natural Language Processing Using Very Large Corpora, Text, Speech and Language Technology, Vol 11. DOI: 10.1007/978-94-017-2390-9_10.
5. **Khoufi, N., Aloulou, C., Belguith, L. H. (2014).** Chunking Arabic texts using conditional random fields. 2014 IEEE/ACS 11th International Conference on Computer Systems and Applications, AICCSA, IEEE, pp. 428–432 DOI: 10.1109/AICCSA.2014.7073230.
6. **Khoufi, N., Boudokhane, M. (2013).** Statistical-based system for morphological annotation of Arabic texts. Proceedings of the Student Research Workshop associated with RANLP 2013, pp. 100–106.
7. **Zitouni, I. (2014).** Natural language processing of Semitic languages, Berlin: Springer. pp. 299–334.
8. **Khoufi, N., Aloulou, C., Belguith, L. H. (2016).** Parsing Arabic using induced probabilistic context free grammar. International Journal of Speech Technology, Vol. 19, pp. 313–323. DOI: 10.1007/s10772-015-9300-x.
9. **Khoufi, N., Aloulou, C., Belguith, L. H. (2015).** Arabic probabilistic context free grammar induction from a treebank. Research in Computing Science, Vol. 90, pp. 77–86.
10. **Debili, F., Achour, H., Souissi, E. (2002).** La langue arabe Et L'ordinateur: de L'étiquetage grammatical à la Voyellation automatique. Correspondances, Vol. 71, pp. 10–28.
11. **Diab, M., Hacioglu, K., Jurafsky, D. (2004).** Automatic tagging of Arabic text: From raw text to base phrase chunks. Proceedings of HLT-NAACL 2004: Short papers, pp. 149–152.
12. **Diab, M. (2007).** Improved Arabic base phrase chunking with a new enriched POS tag set. Proceedings of the 2007 Workshop on

Computational Approaches to Semitic Languages: Common Issues and Resources, pp. 89–96.

13. **Allwein, E. L., Schapire, R. E., Singer, Y. (2000).** Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine Learning Research, Vol. 1, pp. 113–141.
14. **Mohammed, M. A., Omar, N. (2011).** Rule based shallow parser for Arabic language. Journal of Computer Science, Vol. 7, No. 10, pp. 1505–1514. DOI: 10.3844/jcssp.2011. 1505.1514.
15. **Allwein, E. L., Schapire, R. E., Singer, Y. (2000).** Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of machine learning research, Vol. 1, pp. 113–141.
16. **Fraj, F. B., Kessentini, M. (2012).** Combinatorial classification for chunking Arabic texts. International Journal of Artificial Intelligence & Applications, Vol. 3, No. 5, pp. 63–71. DOI: 10.5121/ijaia.2012.3506.
17. **Khoufi, N., Aloulou, C., Hadrich-Belguith, L. (2015).** Enhancing CRF model with N-grams for Arabic texts chunking. The 25th International Business Information Management Conference, IBIMA 2015, pp. 2877–2884.
18. **Maamouri, M., Bies, A., Buckwalter, T., Mekki, W. (2004).** The Penn Arabic treebank: Building a large-scale annotated Arabic corpus. NEMLAR conference on Arabic language resources and tools, Vol. 27, pp. 466–467.
19. **Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013).** Efficient estimation of word representations in vector space. DOI: 10.485 50/arXiv.13 01.3781.
20. **Barhoumi, A., Camelin, N., Aloulou, C., Estève, Y., Belguith, L. H. (2019).** An empirical evaluation of Arabic-specific embeddings for sentiment analysis. International Conference on Arabic Language Processing, Springer, Cham. pp. 34–48.
21. **Sahidullah, M., Patino, J., Cornell, S., Yin, R., Sivasankaran, S., Bredin, H., Korshunov, P., Brutti, A., Serizel, R., Vincent, E., Evans, N., Marcel, S., Squartini, S., Barras, C. (2019).** The speed submission to DIHARD II: Contributions & lessons learned. arXiv preprint arXiv:1911.02388. DOI: 10.485 50/arXiv.1911.02388.
22. **Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., Schmidhuber, J. (2009).** A novel connectionist system for improved unconstrained handwriting recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 3, No. 5.
23. **Sak, H., Senior, A. W., Beaufays, F. (2014).** Long short-term memory recurrent neural network architectures for large scale acoustic modeling. INTERSPEECH 2014, pp. 338–348.
24. **Sundermeyer, M., Alkhouli, T., Wuebker, J., Ney, H. (2014).** Translation modeling with bidirectional recurrent neural networks. Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP, pp. 14–25.
25. **Cho, K., van-Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014).** Learning phrase representations using RNN encoder-decoder for statistical machine translation. DOI: 10.48550/arXiv.1406.1078.
26. **Gers, F. A., Schmidhuber, J., Cummins, F. (2000).** Learning to forget: Continual prediction with LSTM. Neural computation, Vol. 12, No. 10, pp. 2451–2471. DOI: 10.1162/0899766 00300015015.
27. **Ravanelli, M., Brakel, P., Omologo, M., Bengio, Y. (2018).** Light gated recurrent units for speech recognition. IEEE Transactions on Emerging Topics in Computational Intelligence, Vol. 2, No. 2, pp. 92–102. DOI: 10.1109/TETCI.2017.2762739.
28. **Gruber, N., Jockisch, A. (2020).** Are GRU cells more specific and LSTM cells more sensitive in motive classification of text? Frontiers in artificial intelligence, Vol. 3, No. 40. DOI: 10.3389/frai.2020.00040.

Article received on 27/03/2023; accepted on 22/04/2024.
**Corresponding author is Nabil Khoufi.*

ISSN 2007-9737

Deep Learning-Based Text Classification to Improve Web Service Discovery

Hadj Madani Meghazi[1,2,3,*], Sid Ahmed Mostefaoui[1,2], Moustafa Maaskri[1,2], Youcef Aklouf[3]

[1] University of Tiaret, Tiaret,
Algeria

[2] University of Tiaret, Laboratory of Research in Artificial Intelligence and Systems,
Algeria

[3] University of Science and Technology Houari Boumediene,
Algiers

{h.meghazi, s_mostefaoui, moustafa.maaskri}@univ-tiaret.dz, yaklouf@usthb.dz

Abstract. Due to the rising number of firms and organizations offering access to their business data or resources on the internet through APIs, there has been a significant increase in the number of web APIs. This poses a difficulty in swiftly and effectively finding online APIs. In order to tackle this problem, the introduction of service classification has been implemented to streamline the process of finding services within a vast array of options. Prior approaches have endeavored to classify web services based on semantic characteristics, although their precision has been constrained. This work introduces a novel strategy named "DeepLAB-WSC" to improve the identification of web services. The approach specifically emphasizes actions derived from textual descriptions of web services and utilizes advanced techniques from deep learning-based text classification. The suggested methodology was evaluated using a real-world web API dataset and achieved superior results compared to existing state-of-the-art research.

Keywords. Service classification, action extraction, text classification, deep learning, web services discovery.

1 Introduction

Service-Oriented-Architecture (SOA) and its primary implementation technology, Web Services (WS), have revolutionized the process of designing and developing corporate applications for software suppliers. Functioning as fundamental units that can transmit and modify data, they interconnect to generate novel composite value-enhanced services that may be accessed as needed.

The most crucial step of the Web services' consumption cycle was and remains service discovery. This is because of their sheer number, which grows exponentially, as well as the fact that better services or mashups will be produced if we can make a good discovery. Research on this subject can be grouped into three (03) categories: syntactic, semantic, and social. The first one covers syntactic techniques for measuring the degree of services similarity that is mostly based on WSDL descriptions, which are published in and processed, word-for-word, by a UDDI directory [21]. Semantic techniques fall within the second category. Several formalisms have been suggested in this context to incorporate a semantic aspect, starting with the straightforward annotation of WSDL descriptions (WSDL-S) [1], moving on to the proposal of a high-level WS ontology (OWL-S) [14], and finally giving birth to a new conceptual model (WSMO) [18]. The third category includes methods that propose a model based on a social network of web services to intercept and exploit the

history of their interactions in order to improve their discovery [13].

To reduce complexity, several studies have been conducted to address this issue, and research has demonstrated that Web service classification/clustering is the optimal approach for not just the discovery process but also for recommendation, selection, and composition. [22, 4, 24, 20]. In this work, we will propose an approach for the classification of web services that falls under the semantic category but without resorting to a domain ontology.

This is not specific to our work, but it is associated with the remarkable success of Deep Learning methods and applications. More precisely, word embedding techniques like Word2Vec [19], Glove [17], or even BERT [3], which are great ways to capture semantics without using ontologies. These latter have considerably slowed down the development of proposals in the field given the extensive work involved in developing high-level ontologies, annotating Web services, and performing related inference processes.

In Our approach, we started by analyzing the textual descriptions of web services in order to extract, with an algorithm, what we will qualify as service actions. Then, we extended and exploited a selection of stat-of-art text classifiers and accentuated their learning with these actions. Finally, we conducted extensive experiments on more than 8,400 real-world web services from ProgrammableWeb[1] to evaluate the effectiveness of our proposal, and we have unequivocally demonstrated that our suggested technique can get a higher level of precision in classification and outperform the most advanced methods currently available. The subsequent sections of this work are structured in the following manner.

Section 2 provides a comprehensive review of relevant research, with a specific emphasis on the application of Deep Learning methods for the classification of Web services. Section 3 is specifically devoted to showcasing our methodology. Section 4 provides a comprehensive examination and interpretation of the experimental findings. Section 5 serves as the final section of the report, providing a conclusion and addressing potential future research.

[1] www.programmableweb.com

2 Related Work

The emergence of web APIs has captured the attention of the academic community and has witnessed substantial engagement. The main sources of data for researching Web services categorization using Deep Learning techniques are Web services description documents and data collected from their environment [26].

For instance, in [28] Cao et al. proposed a Wide and Bi-LSTM model combining all the discrete features in the description documents of Web services and performing the breadth prediction of Web service category to automatically extract the most pertinent semantic information from a Web service document.

Then, a Bi-LSTM architecture uses a topical attention-mechanism to mine the word order and context information of the words in the Web service description documents in order to do Web service classification prediction. In [32], the authors provide a DeepWSC framework for web service clustering that is heuristics-based.

In order to cluster web services effectively, within this method, a signed graph convolutional network is used to extract service composability characteristics from service invocation associations, and an upgraded recurrent convolutional neural network (RCNN) [10] is used to extract deep semantic features from service descriptions.

This is an extension of paper [31], the key difference is that the new DeepWSC utilizes the composability characteristics of services, which are then inputted into the deep neural network with the deep semantic features of web services. These features are combined using an approach to generate integrated implicit properties of web services.

The paper [26] introduces ServeNet, a deep neural network that can automatically extract high-level features from service names and descriptions without the need for feature engineering or length restrictions. ServeNet is

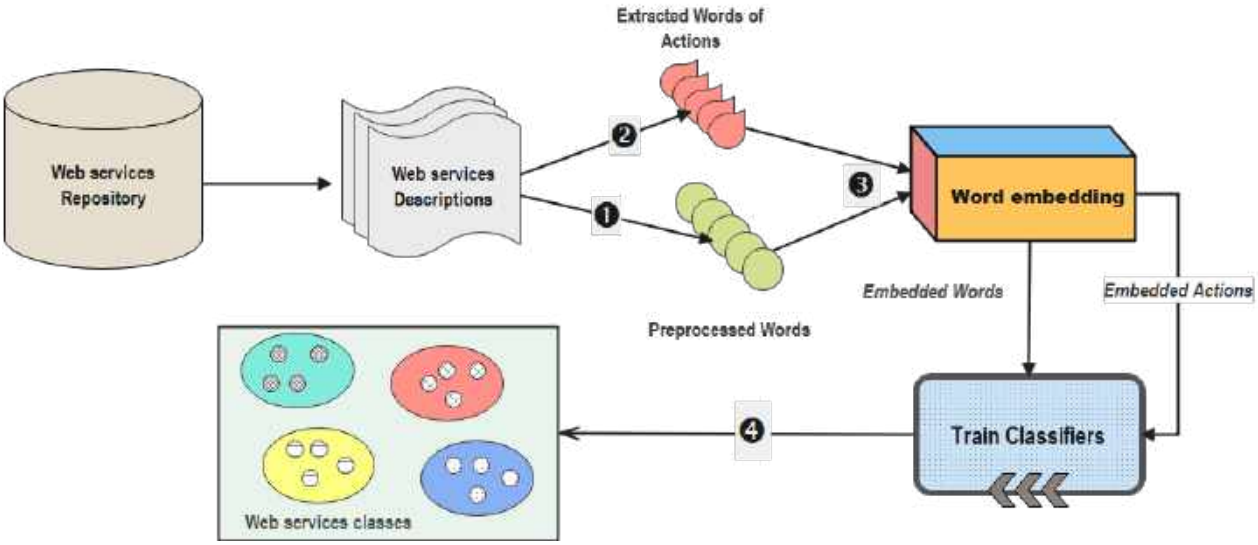


Fig. 1. Overall framework of our DeepLAB-WSC

capable of predicting the classification of services into 50 different categories.

In their study, Kang et al. [9] utilize the attention mechanism to combine the local implicit state vector of a Bidirectional Long Short-Term Memory Network (BiLSTM) with the global Hierarchical Dirichlet Process (HDP) topic vector. They propose a topical attention-based BiLSTM technique for classifying Web services.

The BiLSTM model is designed for the purpose of automatically acquiring the keyword feature representations of Web services. The topic vectors of Web service documents are acquired using HDP during offline training.

A topic attention technique is employed to improve the feature representation by discerning the significance or weight of different keywords in Web service documents.

An attention-based BiLSTM model is also coupled with Information Gain theory to present a Web service classification in [29]. The suggested technique focuses on intricate elements inherent in Web services, such as the significance of various words and the sequential semantic connections between words, through the utilization of IG theory and the attention-based BiLSTM model.

The authors of [23] have developed a new deep neural network that combines a Graph-Convolutional-Network (GCN) and a Bidirectional-Long-Short-Term-Memory (Bi-LSTM) network. This network aims to automatically extract function-description-documents by capturing different relationships in graphs and exploiting them. The GCN is used to extract global spatial features, while the Bi-LSTM network learns sequential features.

Also, Peng et al. [16] present a graph attention network-based Web services classification approach. To begin, it capitalizes on description documents, Web service tags, and the call relationship between mashups and services to create a service relationship network based on Web service composition and shared annotations. The self-attention mechanism then determines the attention coefficient of each service node in the network, and different service nodes in nearby areas are allocated different weights to classify Web services. The graph attention network combines a Web service's content characteristics with structure information.

X Yong et al. introduce LDNM [25], a comprehensive framework for classifying web services. This framework utilizes a deep

Algorithm 1: Extract Actions

Input: Web service textual description
Output: List of Actions[ ]

```
function SGET_ACTION(Sentence)
    if (Item is Verb) and (has dobj) then
        Concatenate the verb with dobj;
        return as Action;
    if (Item is Verb) and (has conjuction) then
        Go to the conjunction item;
        Once you reach the Noun: concatenate the Verb with the Noun;
        if this Noun has conjuction then
            Concatenate also the Verb with the next Noun;
    return Actions;
procedure GET_ACTION(Sentence)
    if (Item is Verb) and (has xcomp) then
        go to the xcomp node;
        concatenate the Verb with each action of SGet_Action(xcomp);
        Add all as Actions to the list of Actions[ ];
    else
        Add all Actions of SGet_Action(Sentence) to the list of Actions[ ];
for each sentence in Web service textual description do
    Get_Action(Sentence);
```

fusion technique to combine structured and unstructured characteristics. The initial step included transforming each service document into a feature vector using two different document representation techniques: topic distribution based on LDA (Latent Dirichlet Allocation) [2] and Doc2vec [11], which is a document embedding model based on neural networks.

Then, using Node2vec [6], they obtained structured representation vectors extracted from service invoking and tagging graphs. Finally, they employ an MLP neural network to fuse these features and train a service classifier.

The methods we just discussed attempt to classify web services based on their textual descriptions or by combining them with other features. When we looked more closely at these methods, we discovered that they treat these descriptions as a collection of words rather than taking into account and differentiating between significant words and their placements in these descriptions.

Original Tags	Stem Tags
(‘Service’, ‘proper noun’)	(‘servic’, ‘adjective’)
(‘used’, ‘**verb**’)	(‘use’, ‘adjective’)
(‘validate’, ‘**verb**’)	(‘valid’, ‘adjective’)

Furthermore, some techniques simply use the stems of words, which can distort the meaning. In this light, we extracted the actions of web services from their textual descriptions using an algorithm, which we will describe in more detail in the following section.

Then, a number of models from the ”Text Classification” [8, 10, 3, 30, 12] domain are extended and adjusted to better fit the classification of these services using these actions as additional features.

3 The Classification Process of the Proposed Approach

The process of our approach, which we named DeepLAB-WSC (Deep Learning Actions Based Web Service Classification), is illustrated in Figure 1. It consists of four (04) steps.

3.1 Step 1: Pre-processing of Web Services Descriptions

Web Services are characterized by their capacity to execute actions and perform tasks. Nevertheless, in the majority of situations, when it comes to presenting information about them, we can only offer a concise written explanation.

In order to effectively identify them using this method, the key question to ask is: "Which specific words or phrases in these descriptions accurately convey the essence of a service?" Initially, we believed that when we encounter a text, the acts described within it are inherently represented by 'verbs'. We began the process by extracting all possible verbs from web service descriptions using pre-existing natural language models that had been trained beforehand.

At this juncture, we encountered two predicaments: The primary concern is that these models lack the ability to precisely identify all verbs. In the statement "Service X searches for all worldwide airlines that operate in a given country", the term 'searches' is identified as a noun. If we isolate each word and disregard its context, what would happen?

The technique becomes more challenging due to the loss of start tags for many words throughout the text cleaning and pre-processing steps, especially after the Stemming step. Consequently, verbs are often misidentified as nouns, adjectives, and so on. Take into account the statement that follows: The words 'Service', 'used', and 'validate' in the statement "Service X is used to validate monetary transactions." are marked as follows:

In most cases, this has the effect of changing the semantic meaning of each word and, as a result, the overall meaning of the description. Because of this, we attempted to overcome this in our situation from the very beginning of our approach. That's why we begin by softly pre-processing the service descriptions in order to retain as many words as possible with their original tags.

3.2 Step 2: Obtain Actions Via Web Service Descriptions

Another problem we faced was the scarcity of verbs in the majority of brief descriptions, making it challenging to utilize these models effectively. In order to tackle this issue, we employed the 'Extract-actions' algorithm at this phase to extract activities as per our definition where xcomp denotes an open-clausal-complement and **dobj** symbolizes the direct object. The algorithm is demonstrated in the following examples:

- **Case 1:**
  - Description: Service X **annotates** text.
  - Actions: [**annotates** text].
- **Case 2:**
  - Description: Service X **annotates** text and images.
  - Actions: [**annotates** text, **annotates** images].
- **Case 3:**
  - Description: Service X uses Twitter API to track followers.
  - Actions: [**utilizes** Twitter, **utilizes track** followers].

3.3 Step 3: Actions and Descriptions Embedding

Of the five (05) classification models we trained (see Section 3.3), we used two methods for the Word embedding process: Glove and BERT.

- GloVe (Global-Vectors for word representation) [17] is an unsupervised learning method developed by Stanford University researchers with the goal of generating Word Embeddings by aggregating global word co-occurrence matrices from a given corpus.

  We used the variant with a length equal to 100 and trained on 6B tokens including 400K

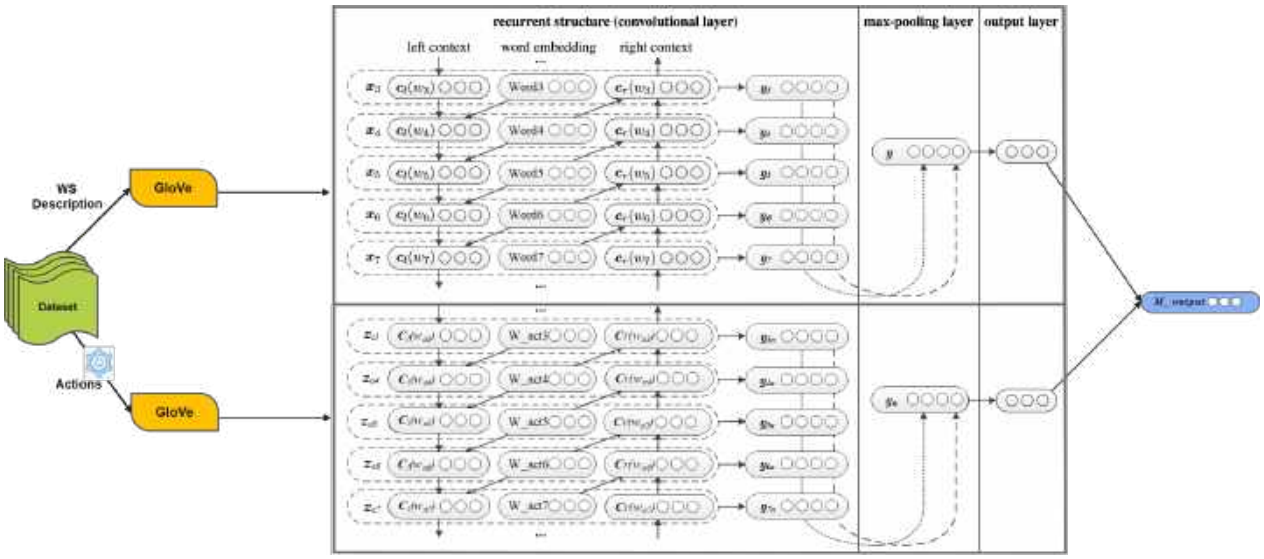


Fig. 2. WSC2RCNN architecture [7]

vocab. This variant is used on the WSC2RCNN and TextING-Based [30]classification models.

– BERT(Bidirectional-Encoder-Representations from Transformers) [3] is a state-of-the-art language representation model.

It is a huge deep bidirectional encoder-based transformer model that has been pre-trained on more than 110 million parameters. BERT is used as the word embedding model for our BERT and BERT-GCN based [12] classification models.

3.4 Step 4: Classification based on Services' Actions and Descriptions Embedding

Our approach's learning process consists of training our classifiers using embedding representations of web service descriptions, then attempting to accentuate their learning with the actions extracted in Step **2** and embedded in Step **3**. After making a connection between the work seen in Section 2 and the results obtained by [15] and [5], the tested models are:

3.4.1 WSC2RCNN

In our previous work [7], we used a Web Service Classifier with two (02) RCNN deep neural networks. We gave, pre-trained GloVe model, representations for both descriptions and collected actions into a pair of Recurrent-Convolutional-Neural Networks (RCNN) that would be trained.

For both networks, we employed the original version of RCNN [10], which seeks to capture textual semantics by considering sequence word order. This approach utilizes both Recurrent Neural Networks (RNN) to understand the local context of tokens and Convolutional Neural Networks (CNN) to capture long-term dependencies. The left and right contexts of a word w_i, which we determined using the following equations, were represented by $c_l(w_i)$ and $c_r(w_i)$, respectively, for the first RCNN network:

$$c_l(w_i) = f(W^{(l)}c_l(w_{i-1}) + W^{(sl)}e(w_{i-1})), \quad (1)$$

$$c_r(w_i) = f(W^{(r)}c_r(w_{i+1}) + W^{(sr)}e(w_{i+1})), \quad (2)$$

where the activation function f is non-linear. A word's word embedding vector is represented by $e(w)$. The context is transformed into the following hidden layer using a matrix called $W^{(l)}$. $W^{(sl)}$ is a

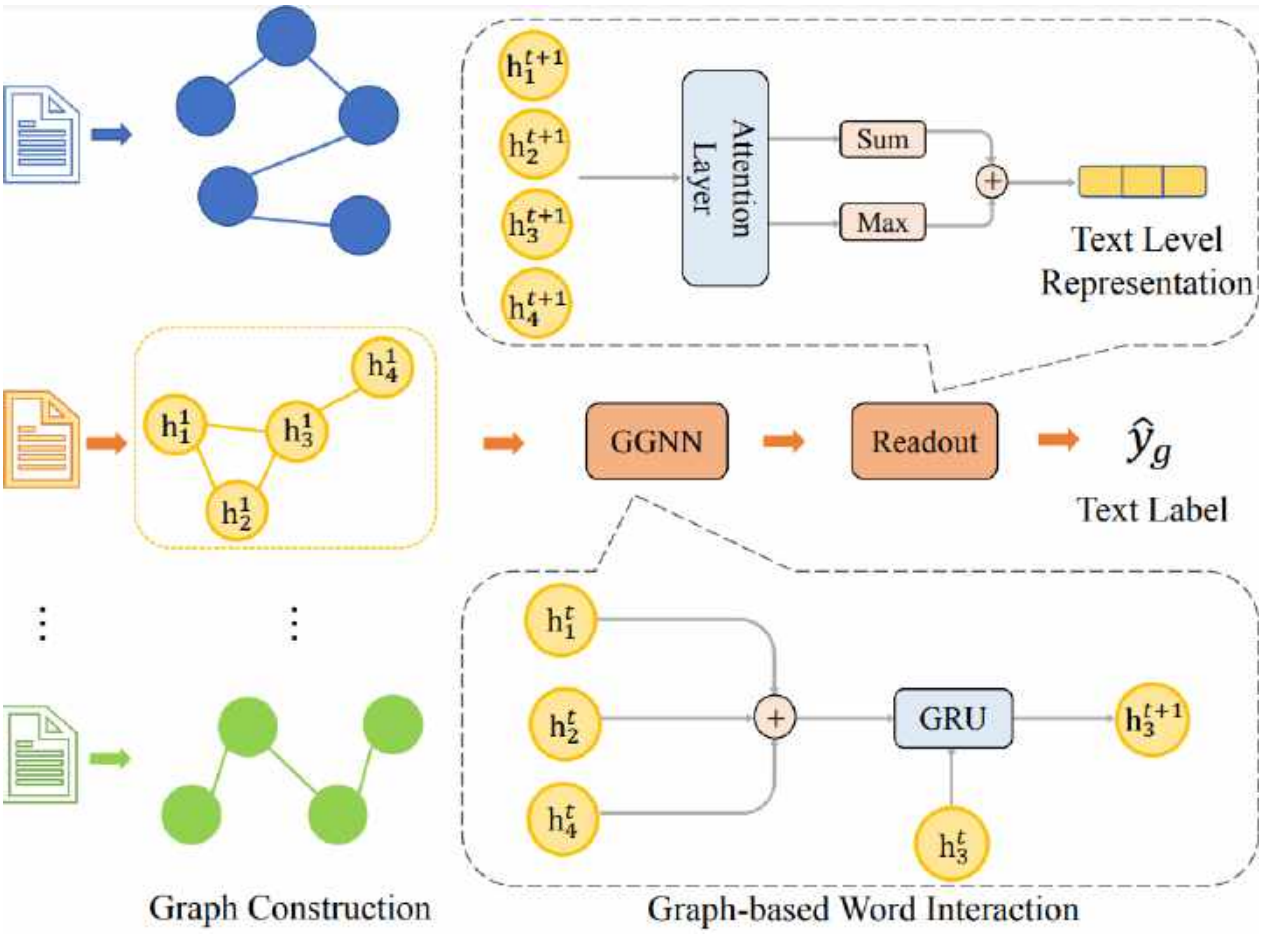


Fig. 3. TextING architecture [30]

matrix that links the semantic of the present word to the left context of the word that comes after it.

Equation (3) is used to obtain the latent semantic representation of the y_i vector:

$$y_i = \tanh(Wx_i + b), \tag{3}$$

where a word w_i is represented by x_i:

$$x_i = [c_l(w_i); e(w_i); c_r(w_i)]. \tag{4}$$

A similar calculation is made for y_a (of actions) for the second network, but it is applied to the words of the extracted actions. The 20 key categories of high-quality Web services from the dataset are linked to the outputs of both RCNN networks, which are goals of WSC2RCNN. The final output is then prepared by summing the outputs from the two RCNN networks.

A softmax activation layer is used to normalize the final output, M_output, because we have a multi-class issue. We present our model in Figure 2.

3.4.2 TextING Classifier

We conducted experiments on WS descriptions using TextING. The authors of [30] provide a novel graph neural network for text categorization. In this network, each document is treated as a separate graph, allowing for the learning of word interactions at the text level. In this process, distinct graphs are generated for each text, and

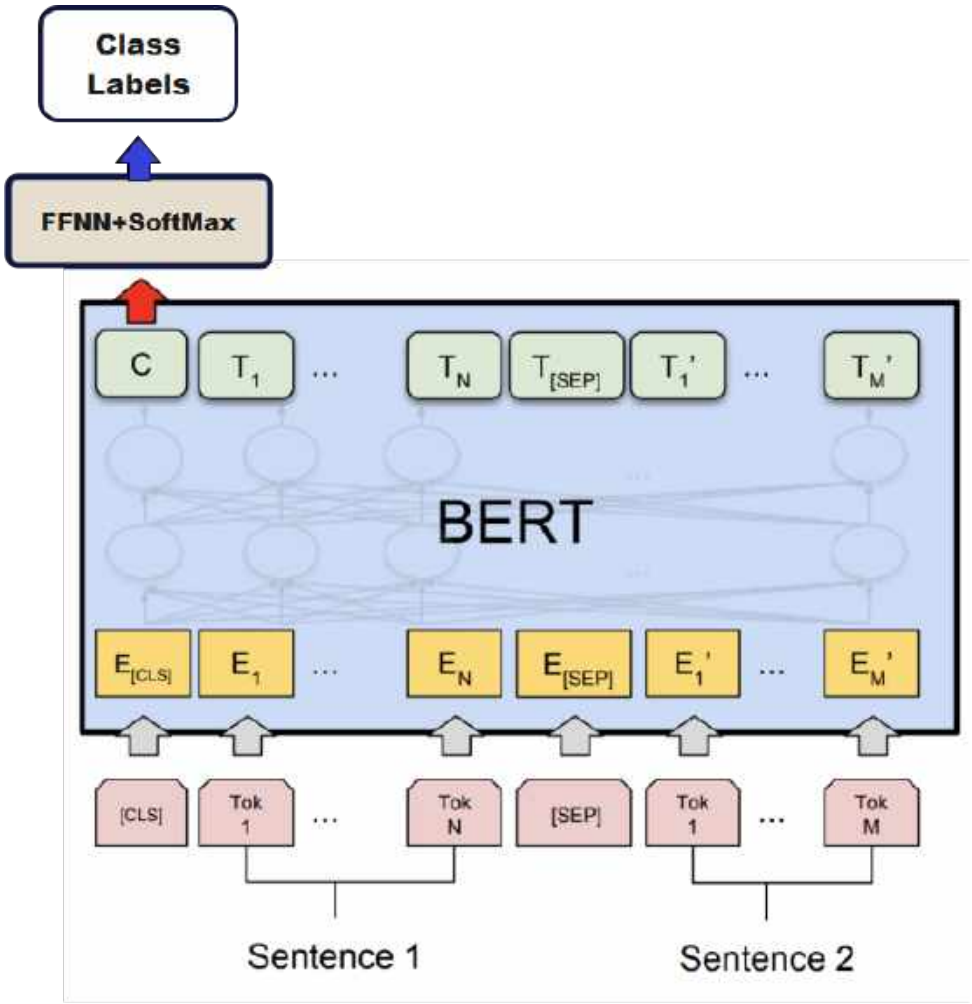


Fig. 4. BERT classifier

subsequently, Graph Neural Networks (GNN) are employed to acquire detailed word representations by considering their local structures. Additionally, GNN has the capability to produce embeddings for words that have not been seen before in the new document. Ultimately, the word nodes are included into the document embedding.

3.4.3 BERT Classifier

A fine tuned BERT is used and a number of layers are added to elaborate a BERT Web Service classifier, shown in figure 4. The model used is bert-base-uncased, which consists of 12 layers, 768 hidden units, 12 attention heads, and a total of 110 million parameters, and a softmax activation layer to normalize the final output.

3.4.4 BERT_With_Actions Classifier

We made multiple attempts to incorporate the extracted actions efficiently while trying to increase the performance of the “BERT Classifier” and validate our approach. The solution found consists of concatenating the sentences of the web service descriptions with the actions repeated “Rep” times

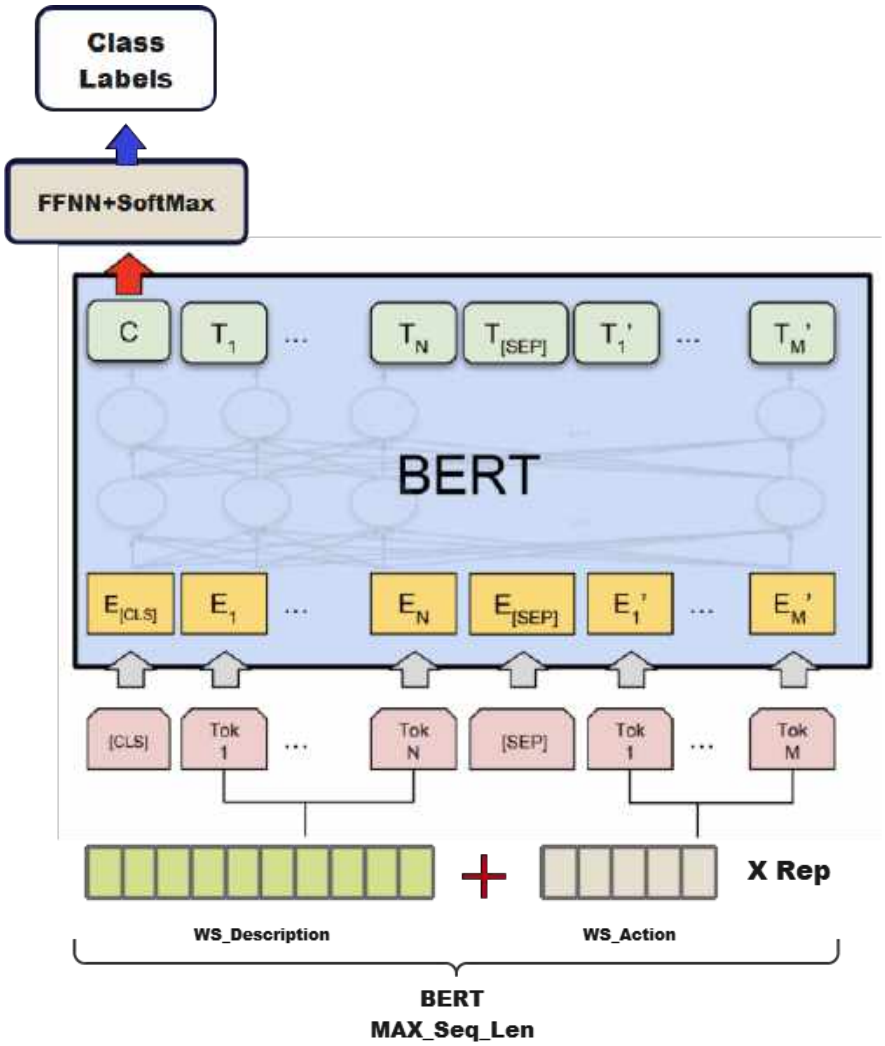


Fig. 5. BERT + Actions classifier

while not exceeding the BERT variant's maximum sequence length (See figure 5):

$$\text{Input}_{\text{BERT\_with\_Actions}} = \text{WS\_Description} + \text{WS\_Actions} \times \text{Rep}. \quad (5)$$

3.4.5 Classifiers based on BERT-GCN

We also put Yuxiao Lin et al.'s proposition (BERT-GCN) to the test. In [12], BERT with his large-scale pretraining, and GCN [27] with his transductive learning, are trained together.

First, a graph with both word nodes and document nodes is built. The weight of an edge that connects two nodes i and j is defined in the following manner:

$$A_{i,j} = \begin{cases} \text{PPMI}(i,j), & i, j \text{ are words and } i \neq j, \\ \text{TF} - \text{IDF}(i,j), & i \text{ is a document, } j \text{ is a word,} \\ 1, & i = j, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The word-document edges and word-word edges are determined using the term frequency-inverse document frequency (TF-IDF)

Table 1. The allocation of Web services across selected categories

ID	Primary Cat	# of services	ID	Primary Cat	# of services
0	Tools	887	10	Telephony	342
1	Financial	757	11	Security	312
2	Messaging	591	12	Reference	304
3	eCommerce	553	13	Email	299
4	Payments	553	14	Search	290
5	Social	510	15	Travel	294
6	Enterprise	509	16	Video	281
7	Mapping	429	17	Education	277
8	Government	371	18	Advertising	274
9	Science	357	19	Transportation	269

and positive point-wise mutual information (PPMI) measures, respectively.

Then, the GCN layers iteratively update document nodes after initializing them using BERT-style embeddings, and the main training aim is to do linear interpolation between the BertGCN prediction and the BERT prediction. This may be expressed as:

$$Z = \lambda\, Z_{\mathrm{GCN}} + (1-\lambda)\, Z_{\mathrm{BERT}}, \tag{7}$$

where λ regulates the tradeoff between the two models. The same process of 3.4.4 is applied to BERT_GCN_With_Actions Classifier in terms of the integration of actions while respecting the maximum length imposed by BERT used variant.

4 Experiments

4.1 Used Evaluation Metrics

The evaluation of selected models has been conducted using four widely used evaluation measures: Purity, NMI, Recall, and F_1-measure. **Purity** is a metric used for guided cluster validation. It is calculated using the following formula:

$$\mathrm{Purity}(\Omega,\, \mathbb{C}) = \frac{1}{N}\sum_{i=1}^{k} \max_{j} \left|\omega_i \cap c_j\right|, \tag{8}$$

where $\Omega = \{w_1, w_2, ..., w_K\}$ represents the collection of clusters of web services, whereas $\mathbb{C} = \{c_1, c_2, ..., c_J\}$ represents the collection of classes of web services. **NMI** is a metric that relies on mutual information and is precisely defined as:

$$\mathrm{NMI}(\Omega,\, \mathbb{C}) = \frac{2 \times I(\Omega;\, \mathbb{C})}{H(\Omega) + H(\mathbb{C})}, \tag{9}$$

where can we obtain the mutual information I using:

$$I(\Omega;\, \mathbb{C}) = \sum_{i=1}^{k}\sum_{j=1}^{k} P(\omega_i \cap c_j) \log \frac{P(\omega_i \cap c_j)}{P(\omega_i) \cap P(c_j)}. \tag{10}$$

And the entropy H using:

$$H(\Omega) = -\sum_{i=1}^{k} P(\omega_i) \log P(\omega_i). \tag{11}$$

Recall is a metric used to determine the accuracy of predicting the real class by measuring the proportion of properly predicted instances. It is computed as follows:

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{12}$$

where TP represents the count of services properly allocated to their respective class, and FN represents the count of services where the model wrongly forecasts their positive class as negative.

Table 2. The classification performance of tested methods

Models	Purity	NMI	Recall	F1-Score
DeepWSC	0.5708	0.4856	0.3821	0.3969
DeepWSC + Heuristics	0.6379	0.5273	0.4186	0.4356
RCNN*	0.6438	0.5704	0.6438	0.6247
WSC2RCNN	**0.6595**	**0.5795**	**0.6588**	**0.6512**
TextING	0.6887	0.5986	0.6864	0.6714
BERT	0.7746	0.6924	0.7746	0.7712
BERT + Actions	**0.7785**	**0.6931**	**0.7786**	**0.7759**
BERT-GCN	0.7788	0.6935	0.7788	0.7717
BERT-GCN + Actions	**0.7810**	**0.6972**	**0.7811**	**0.7745**

The last used measure is $\mathbf{F}_1$ which is calculated using the next formulas:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{13}$$

The term "FP" represents the count of APIs where the model wrongly forecasts their negative classes as positive:

$$\text{F}_1\text{-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{14}$$

4.2 Experimental Environment

We performed tests to assess and showcase the efficiency of our methodology. All experiments were executed on a platform equipped with an Intel (R) Xeon (R) platinum 8259CL CPU@2.50GHz (32 cores) and 256GB RAM.

Utilizing the Dataset[2] of [31, 32] from ProgrammableWeb, the leading online registry for Web services with a vast collection of over 23,000 APIs, we accessed a compilation of 17,923 authentic web services that were obtained by web crawling. The experimental data comprises 8,459 high-quality WS from the top 20 classes (refer to Figure 1).

[2]github.com/aourhtnowvherlcaer/programmableWeb

4.3 Experimental Results and Discussions

We compared our results to those of [31, 32] in order to correctly interpret them. Table 2 summarizes all of the results obtained by the various studied methods. We started by putting in tests for RCNN, TextING 3.4.2, BERT 3.4.3, and BERT-GCN 3.4.5 classifiers. The results were satisfactory, starting with the RCNN classifier, which alone exceeded those of [32] on all metrics.

This is because the authors of [31] wanted to get closer to the semantics of WS descriptions by trying to capture the context of each word with an RCNN deep neural network. However, by applying strict stemming, the meaning of many words is lost (See Section 3.1).

This is what we could observe in their pre-processed dataset. This issue has been fixed and the results of our RCNN classifier exceeded the best of [32]'s by an average of 26.57% on all the evaluation metrics.

The first tested method of the GNN class (TextING) achieves an average improvement of 34.89%. On [32], BERT and BERT-GCN had average advantages of 53.7% and 54.2%, respectively. To complete the evaluation of our approach's effectiveness, each of the proposed classifiers has gone through an extra training process in which we have attempted to include the actions retrieved by our "**Extract-Actions**" algorithm (See Section 3.2).

The results have demonstrated that the classifiers' performance improves each time these actions are combined. Our WSC2RCNN Classifier outperforms our RCNN and presents an average advantage of 5.51% across all the evaluation metrics. Furthermore, our BERT+Actions classifier achieves an average advantage of 0.43% over the BERT classifier. Ending with BERT-GCN+Actions, which outperforms all classifiers seen in this work and shows a 0.37% improvement over BERT-GCN.

5 Conclusion and Future Research

This work introduces a novel method for classifying web services, named DeepLAB-WSC, that relies on the concise textual descriptions of the services. Our methodology stands out by prioritizing the activities executed by web services, which are derived from their descriptions, and use deep learning text classification techniques to categorize the services.

Our comparison trials have demonstrated that DeepLAB-WSC surpasses current cutting-edge techniques for classifying web services in terms of all performance parameters. Our technique has a key benefit in that it specifically targets the most crucial aspect of the description, namely the actions, in order to enhance the accuracy of categorization.

Looking ahead, we plan to extend this work by leveraging the power of BERT and GNN to build social networks of web services based on the similarities (BERT based classifiers) and complementarities (predicting the next sentence/document) of their descriptions. Making this social dimension profitable will allow usto enhance both the process of finding and combining online services.

References

1. **Akkiraju, R., Farrell, J., Miller, J. A., Nagarajan, M., Sheth, A. P., Verma, K. (2005).** Web service semantics - WSDL-S. www.w3.org/submissions/WSDL-S/.
2. **Blei, D. M., Ng, A. Y., Jordan, M. I. (2003).** Latent dirichlet allocation. The Journal of Machine Learning Research, Vol. 3, pp. 993–1022.
3. **Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2019).** BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologie, pp. 4171–4186. DOI: 10.18653/v1/n19-1423.
4. **Elgazzar, K., Hassan, A. E., Martin, P. (2010).** Clustering WSDL documents to bootstrap the discovery of web services. IEEE International Conference on Web Services, pp. 147–154. DOI: 10.1109/icws.2010.31.
5. **Gasparetto, A., Marcuzzo, M., Zangari, A., Albarelli, A. (2022).** A survey on text classification algorithms: From text to predictions. Information, Vol. 13, No. 2, pp. 83. DOI: 10.3390/info13020083.
6. **Grover, A., Leskovec, J. (2016).** node2vec: Scalable feature learning for networks. Proceedings of the 22nd ACM Special Interest Group on Knowledge Discovery and Data Mining and International Conference on Knowledge Discovery and Data Mining, pp. 855–864. DOI: 10.1145/2939672.2939754.
7. **Hadj-Madani, M., Youcef, A. (2022).** WSC2RCNN: A deep learning actions-based classifier for improved web service discovery. Computación y Sistemas, Vol. 26, No. 4. DOI: 10.13053/cys-26-4-4069.
8. **Joachims, T. (1998).** Text categorization with support vector machines: Learning with many relevant features. Proceedings of the European Conference on Machine Learning, pp. 137–142. DOI: 10.1007/bfb0026683.
9. **Kang, G., Xiao, Y., Liu, J., Cao, Y., Cao, B., Zhang, X., Ding, L. (2021).** Tatt-BiLSTM: Web service classification with topical attention-based BiLSTM. Concurrency

and Computation: Practice and Experience, Vol. 33, No. 16. DOI: 10.1002/cpe.6287.

10. **Lai, S., Xu, L., Liu, K., Zhao, J. (2015).** Recurrent convolutional neural networks for text classification. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, No. 1, pp. 2267–2273.
11. **Lau, J. H., Baldwin, T. (2016).** An empirical evaluation of doc2vec with practical insights into document embedding generation. Proceedings of the 1st Workshop on Representation Learning for NLP, pp. 78–86. DOI: 10.18653/v1/w16-1609.
12. **Lin, Y., Meng, Y., Sun, X., Han, Q., Kuang, K., Li, J., Wu, F. (2021).** BertGCN: Transductive text classification by combining GNN and BERT. Proceedings of the Association for Computational Linguistics International Joint Conference on Natural Language Processing. DOI: 10.18653/v1/2021.findings-acl.126.
13. **Maamar, Z., Wives, L. K., Badr, Y., Elnaffar, S., Boukadi, K., Faci, N. (2011).** LinkedWS: A novel web services discovery model based on the metaphor of "social networks". Simulation Modelling Practice and Theory, Vol. 19, No. 1, pp. 121–132. DOI: 10.1016/j.simpat.2010.06.018.
14. **Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K. (2004).** Owl-s: Semantic markup for web services. www.w3.org/submissions/OWL-S/.
15. **Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., Gao, J. (2021).** Deep learning–based text classification: A comprehensive review. ACM Computing Surveys, Vol. 54, No. 3, pp. 1–40. DOI: 10.1145/3439726.
16. **Peng, M., Cao, B., Chen, J., Liu, J., Li, B. (2021).** SC-GAT: Web services classification based on graph attention network. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol. 349, pp. 513–529. DOI: 10.1007/978-3-030-67537-0_31.
17. **Pennington, J., Socher, R., Manning, C. (2014).** Glove: Global vectors for word representation. Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1532–1543. DOI: 10.3115/v1/d14-1162.
18. **Roman, D., de-Bruijn, J., Mocan, A., Lausen, H., Domingue, J., Bussler, C., Fensel, D. (2006).** WWW: WSMO, WSML, and WSMX in a nutshell. Proceedings of the Asian Semantic Web Conference, pp. 516–522. DOI: 10.1007/11836025_49.
19. **Rong, X. (2014).** word2vec parameter learning explained. arXiV. DOI: 10.48550/ARXIV.1411.2738.
20. **Shi, M., Tang, Y., Liu, J. (2019).** Functional and contextual attention-based LSTM for service recommendation in mashup creation. IEEE Transactions on Parallel and Distributed Systems, Vol. 30, No. 5, pp. 1077–1090. DOI: 10.1109/tpds.2018.2877363.
21. **Walsh, A. E. (2002).** UDDI, SOAP, and WSDL: The web services specification reference book. Prentice Hall Professional Technical Reference.
22. **Wang, H., Shi, Y., Zhou, X., Zhou, Q., Shao, S., Bouguettaya, A. (2010).** Web service classification using support vector machine. Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence, pp. 3–6. DOI: 10.1109/ictai.2010.9.
23. **Wang, X., Liu, J., Liu, X., Cui, X., Wu, H. (2020).** A spatial and sequential combined method for web service classification. Proceedings of the Asia Pacific Web and Web-Age Information Management Joint International Conference on Web and Big Data, pp. 764–778. DOI: 10.1007/978-3-030-60259-8_56.
24. **Xia, B., Fan, Y., Tan, W., Huang, K., Zhang, J., Wu, C. (2015).** Category-aware API clustering and distributed

recommendation for automatic mashup creation. IEEE Transactions on Services Computing, Vol. 8, No. 5, pp. 674–687. DOI: 10.1109/tsc.2014.2379251.

25. **Xiao, Y., Liu, J., Kang, G., Cao, B. (2021).** LDNM: A general web service classification framework via deep fusion of structured and unstructured features. IEEE Transactions on Network and Service Management, Vol. 18, No. 3, pp. 3858–3872. DOI: 10.1109/tnsm.2021.3084739.
26. **Yang, Y., Qamar, N., Liu, P., Grolinger, K., Wang, W., Li, Z., Liao, Z. (2020).** ServeNet: A deep neural network for web services classification. IEEE International Conference on Web Services, pp. 168–175. DOI: 10.1109/icws49710.2020.00029.
27. **Yao, L., Mao, C., Luo, Y. (2018).** Graph convolutional networks for text classification. Proceedings of the 33rd AAAI Conference on Artificial Intelligence, pp. 7370–7377. DOI: 10.48550/ARXIV.1809.05679.
28. **Ye, H., Cao, B., Peng, Z., Chen, T., Wen, Y., Liu, J. (2019).** Web services classification based on wide and Bi-LSTM model. IEEE Access, Vol. 7, pp. 43697–43706. DOI: 10.1109/access.2019.2907546.
29. **Zhang, X., Liu, J., Cao, B., Shi, M. (2021).** Web service classification based on information gain theory and bidirectional long short-term memory with attention mechanism. Concurrency and Computation: Practice and Experience, Vol. 33, No. 13. DOI: 10.1002/cpe.6202.
30. **Zhang, Y., Yu, X., Cui, Z., Wu, S., Wen, Z., Wang, L. (2020).** Every document owns its structure: Inductive text classification via graph neural networks. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 334–339. DOI: 10.18653/v1/2020.acl-main.31.
31. **Zou, G., Qin, Z., He, Q., Wang, P., Zhang, B., Gan, Y. (2019).** Deepwsc: A novel framework with deep neural network for web service clustering. IEEE International Conference on Web Services, pp. 434–436. DOI: 10.1109/icws.2019.00077.
32. **Zou, G., Qin, Z., He, Q., Wang, P., Zhang, B., Gan, Y. (2022).** DeepWSC: Clustering web services via integrating service composability into deep semantic features. IEEE Transactions on Services Computing, Vol. 15, No. 4, pp. 1940–1953. DOI: 10.1109/tsc.2020.3026188.

Article received on 20/03/2023; accepted on 21/04/2024.
* *Corresponding author is Hadj Madani Meghazi.*

ISSN 2007-9737

A Transfer Learning Approach for Identification of Two-Wheeler Brand and Model

S Mohammed Mansoor Roomi, Lokesh V S*, Shankar Mahadevan G, Priya K

Thiagarajar College of Engineering,
Electronics and Communication Engineering,
India

smmroomi@tce.edu, {vslokesh10, shankarmahadeva12901, priya5586}@gmail.com

Abstract. Motorcycles were originally designed to provide safer, more efficient, and more comfortable rides, but they are now also used for criminal purposes. Due to overspeeding, motorcycles often get involved in accidents, and it is challenging for law enforcement officials to identify the culprits from CCTV footage or spectator accounts. This paper presents a solution to this challenge by using a pre-trained deep learning model to detect, classify, and identify motorcycle models. To overcome the limited availability of annotated bike databases, this proposed work created a new bike dataset that includes 5000 annotated images sourced from major search engines, CCTV footage, and manual captures from bike showrooms. Then the bike brand was identified in 27 classes by the Faster RCNN pre-trained model and achieved an accuracy of 94.35%. The proposed model was compared with the other pre-trained models such as YOLO V5 and MobileDET, among these, the Faster RCNN provided better identification accuracy.

Keywords. Bike dataset, wheel rim, headlamp, yolo V5, MobileDET, faster RCNN, imbalanced learning, and weighted loss learning.

1 Introduction

Two-wheelers are a common form of transportation in India but are also frequently involved in serious accidents and crimes. As a result, research on smart traffic systems, including vehicle detection, identification, enumeration, and traffic statistic estimate, has become increasingly popular. Researching traffic estimates, speed calculations, motorcycle helmet use, vehicle tracking, as well as occlusion analysis requires segmenting automobiles on public roadways.

The proposed work aims to address the challenge of detecting and classifying motorbikes on public roads. As our world moves faster, the number of vehicles on the road is increasing daily, along with a corresponding rise in crimes and accidents involving bikes. This increase appears to be linear.

To find who is the cause of the accident or the culprit in a crime. The investigation officer enquires about the people in and around the place where the incident has taken place. but they won't have recognized or seen the person's face or the number plate they may have seen only what color the bike was and how did the bike look like or the color of the bike.

Hence to find a solution to this problem, the proposed work is to find the bike model using the bike's wheel rim and bike headlamp. we could have used the number plate to identify the bike and trace out the person who rode the bike but sometimes the number plate may be damaged or sometimes the numberplate may be slightly faded and which is not visible at all or sometimes the bike maybe without number plate also hence without wheel rim a bike cannot move or is not complete hence with the help of wheel rim and headlamp we are going to detect the bike model. Figure 1 shows a sample image of the bike with localization of the wheel rim and headlamp.

2 Related Works

Research into computer vision and image processing (IP) techniques for automating vehicle classification has been ongoing.

Fig. 1. Sample images from the collected data set

Fig. 2. Examples of annotated bike categories are presented, demonstrating variations in both form and pose

One possible approach involves pre-processing images using IP techniques to enhance their suitability for input into a machine-learning model. For example, in [1], a stationary camera was used to record footage of vehicles, which were then classified using three levels of processing and appropriate cameran calibration. Similarly, [2] suggested using IP techniques, Invariant Feature Transform (IFT), K-means clustering, and euclidean distance matching to classify vehicles into different categories for Electronic Tolling Collection.

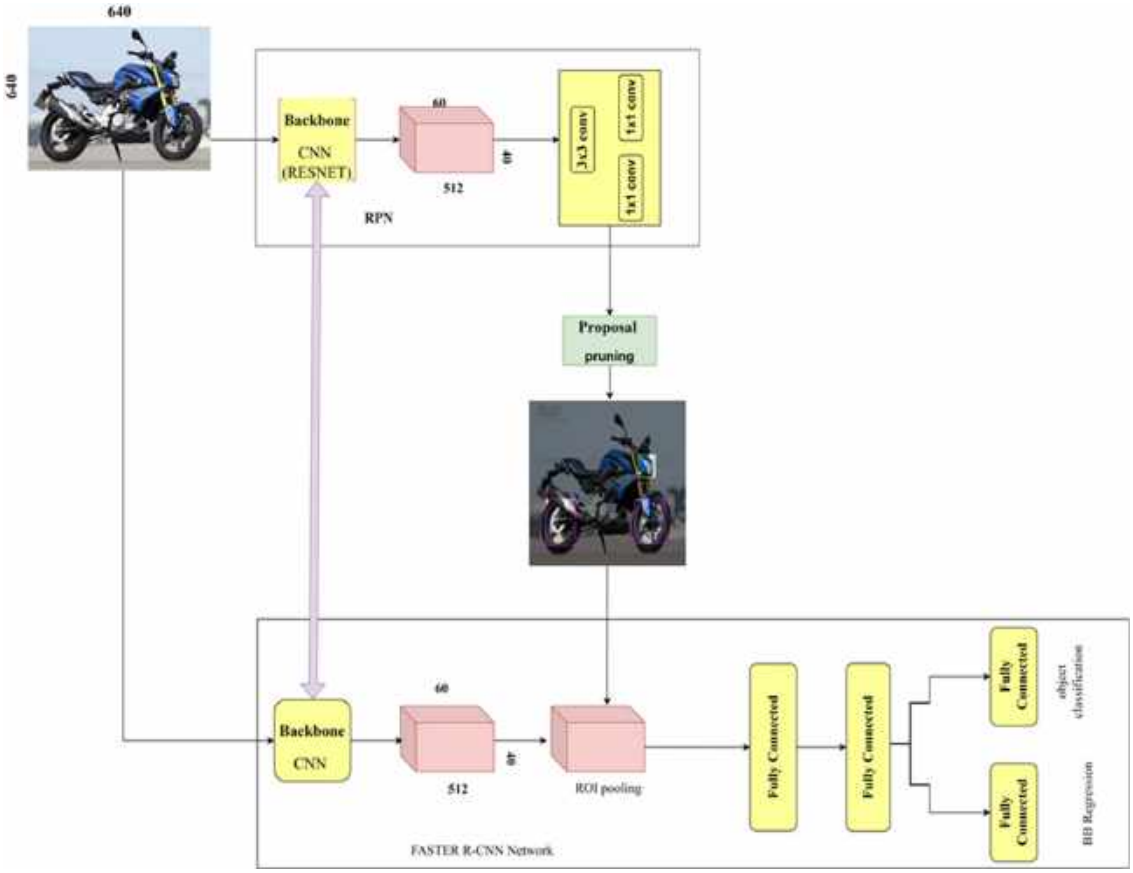


Fig. 4. Flow of faster RCNN architecture

Another example is a pre-crash detection system, as described in [3], which utilized a front-facing low-light camera to detect vehicles and avoid accidents. This system used a combination of multiscale-driven hypothesis generation and appearance-based hypothesis verification to achieve good results in all kinds of weather and road conditions.

In [4], Sonoda et al. proposed a system for detecting moving objects, such as vehicles and pedestrians, at an intersection, which warns drivers of potential dangers. The system employed a mixture of Gaussians for object detection and the Lucas-Kanade Tracker (LKT) algorithm for pedestrian tracking.

In [5], researchers used a camera to capture images of vehicles against a static background, allowing them to calculate vehicle length and width and classify the vehicle as either a Light Motor Vehicle (LMV) or a Heavy Motor Vehicle (HMV). The goal of this research was to determine a suitable toll rate.

In [6], an automated vehicle classification system was proposed to distinguish between motorcycles and sedans, using commercially available light curtains to produce images and two machine learning methods, K Nearest Neighbor (KNN) and decision tree learning, to classify the vehicles. Finally, [7] proposed a computer vision system to detect and segment motorcycles that were partially occluded by other vehicles, using a helmet detection system to identify motorcycles.

However, this approach requires significant information, such as helmet radius, camera angle, and camera height, to function effectively [8] developed a system that can identify vehicles even if they are partially occluded and can continue

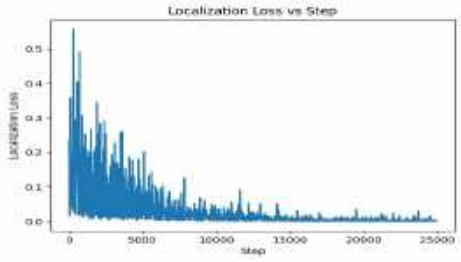


Fig. 5. Plot of localization loss vs step

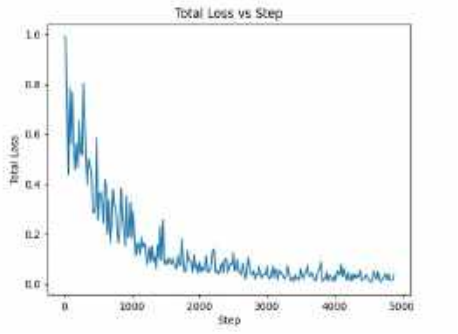


Fig. 6. Plot of total loss vs step

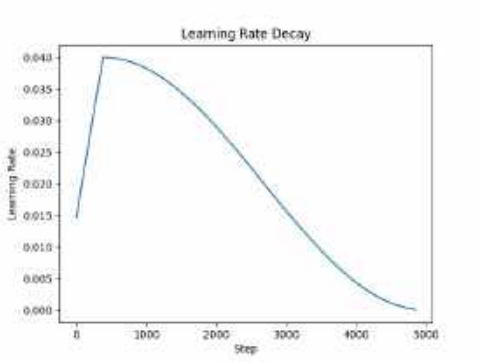


Fig. 7. Plot of learning rate vs step

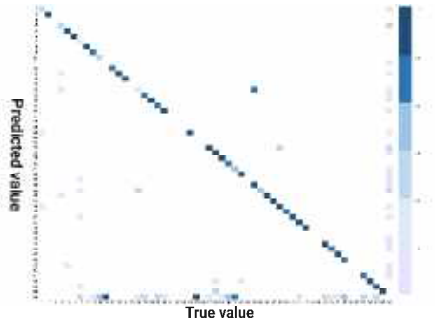


Fig. 8. Confusion matrix of bike model classification

tracking the vehicle despite occlusions. In [9], a computer vision system was developed for detecting bicycles, pedestrians, and motorcycles.

This system used Gabor filtering to detect motion and a Histogram of Oriented Gradient (HOG) descriptors to classify two-wheeled vehicles and pedestrians. Chiverton et al. [10] proposed a system for automatically classifying and tracking motorcycle riders with and without helmets, using histograms derived from the head region of riders to train an SVM classifier.

Although this system achieved high accuracy, the number of test images was limited. The current research work mainly focused on vehicle classification with limited classes and quantity. But there is a need for the bike model dataset with large classes for automation of the bike model identification.

This proposed work created a bike model dataset and pre-trained the Faster Region Convolution Neural Network (RCNN) model for the identification of the bike model. The major contributions of the proposed method are:

– Creation of a bike dataset with 27 classes.
– Proposal of a faster RCNN deep learning model for bike model identification.
– Comparison of proposed work against the existing pre-trained model.

3 Proposed Methodology

The proposed methodology comprises four stages: database creation, augmentation, and training and testing phases to identify motorcycle models using categorical images. The dataset was created by collecting images from various sources LIKE search engines, movies, CCTV footage, and bike showrooms, resulting in a diverse collection of two wheelers.

Data cleaning was performed to remove unwanted images, resulting in a dataset of 1,500 motorcycle images from 27 categories, which were augmented to obtain a total of 5,000 distinct images stored in the database.

However, the target variable showed an imbalance in the number of instances across different classes. Sample images from the created bike dataset are presented in Figure 2. The Faster

Table 1. Labels of bike models

Sl.No	Label	Sample Count
1	BMW g310 r	1,863
2	Ducati panigale	281
3	Gixxer SF	214
4	Hero pleasure+	137
5	Hero splendor +	136
6	Honda Activa	280
7	Honda shine	152
8	Honda Unicorn	260
9	Jawa 42	175
10	Jawa yezdi	53
11	Kawasaki ninja	149
12	Ktm adventure	207
13	Ktm duke	239
14	Ktm RC	57
15	Pulsar ns 200	484
16	Pulsar rs	477
17	Super splendor	132
18	Suzuki Access	287
19	Tork kratos	264
20	Tvs apache rr	468
21	Tvs apache rtr	167
22	Tvs Jupiter	210
23	Tvs ntorq	202
24	Tvs sport	171
25	Yamaha fzs	168
26	Yamaha mt	132
27	Yamaha r15	137

RCNN is a widely used object detection framework that was introduced by Ross Girshick in 2015.

This architecture employs Convolutional Neural Networks (CNNs) for object detection, similar to other popular designs such as You Look Only Once (YOLO) and Single Shot Detector (SSD).

When given an image and bounding boxes as input, the Faster R-CNN network processes the entire image using multiple convolutional and max pooling layers to generate a convolutional feature map.

In this study, the Faster RCNN was used to detect and identify objects in the created bike dataset. To generate feature maps of sizes 60, 40, and 512, the image is passed through the ResNet backbone of the CNN.

For proposal generation, the Region Proposal Network (RPN) is utilized, which benefits from weight sharing between the Faster R-CNN detector backbone and the RPN backbone.

The input image is resized to 640×640 and is fed into the CNN's backbone to initiate the RPN. This study introduces backbone networks like ResNet, which have a network stride of 16, leading to significantly smaller output features compared to the input image. Each point on the output feature map corresponds to two points 16 pixels apart in the input image.

The proposed deep detector network aims to identify whether there is an object present in the input image at each point in the output feature map and estimate its size.

To achieve this, a set of "Anchors" is placed on the input image for each position on the output feature map generated by the backbone network. These anchors point to potential objects of various sizes and aspect ratios that could be present.

The network then determines which of the k-associated anchors in the input image contain objects and adjusts their coordinates to provide bounding boxes as "Object proposals" or areas of interest.

To create 512-d feature maps for each location, the backbone feature map undergoes a 3×3 convolution with 512 units.

Two sibling layers are added: a 1×1 convolution layer for object classification with 18 units and a 1×1 convolution layer for bounding box regression with 36 units. The classification branch's 18 units output a probability of whether an object is present at each location in the backbone feature map.

The regression branch's 36 units produce the size of the object proposals for each point in the backbone feature map. The regression coefficients of each of the 9 anchors for each point in the backbone feature map are obtained using this output.

The coordinates of the anchors containing objects are then adjusted using these regression coefficients

The backbone feature map features are obtained using the ROI pooling layer, which pools them based on RPN-proposed bounding box boundaries. The ROI pooling layer selects the area that matches the proposal from the backbone feature map, divides it into a fixed number of sub-windows, and performs max-pooling over these sub-windows to produce an output of a fixed size.

Table 2. Hyper tuning parameters of the faster RCNN model

Parameter	Value
Optimizer	Adam
Mini Batch Size	4
Device Type	GPU
Learning Rate	0.04
l2Regularization	0.013333
Epoch	2000

Table 3: Faster RCNN accuracy

AveragPrecision (AP)	0.9435
Average Recall (AR)	0.738
Localization loss	0.065122
Classification loss	0.167798
Regularization loss	0.000000
Total loss:	0.275017

The size of the ROI pooling layer's output is (N, 7, 7, 512), where N is the number of proposals generated by the area proposal algorithm.

These features are passed through two fully connected layers before being fed into the sibling classification and regression branches. Two objects need to be labeled in a bike model detection task, and the labels for each object can be either 'headlamp' or 'wheel rim'.

The weights assigned to each label are $W_1 = 0.6$ for 'headlamp' and $W_2 = 0.4$ for 'wheel rim'. We assign a higher weight to the headlamp (0.6) than to the wheel rim (0.4) as the headlamp is more significant than the wheel rim. There are two possible label conditions:

– Both objects have the same label.
– Both objects have different labels.

To determine the label for an object, we use the following process:

– If both objects have the same bike model label, then the label for the object is the same as the label that both objects share.
– If one object has one bike model label and the other has a different bike model label, the proposed work uses the weighted formula $W_1 S_1 + W_2 S_2$ to determine the label for the object. The formula considers the confidence score for each label and the weight assigned to each label. The confidence scores are calculated using a machine learning model, and they reflect the probability of each label is correct.

Then compare the values of W_1S_1 and W_2S_2 to determine which label is more appropriate for the object. If W_1S_1 is greater than W_2S_2, then the object is labeled as the label described by the headlamp, and if W_2S_2 is greater than $W_1 S_1$, then the object is labeled as the label described by the wheel rim. $W_1S_1 \underset{h_1}{\overset{h_0}{\gtrless}} W_2S_2$.

4 Results and Discussions

The results and discussion of the proposed approach are described in this section. The collection of bike images that were developed was divided into an 80:20 ratio. 80 percent of the samples were chosen for training, 10 percent for validation, and the rest for testing.

The proposed algorithm was trained using Google Colab, which was set up on an HP server with 8GB of RAM. The training images were reduced in size at the pre-processing stage to 640×640. The Faster RCNN pre-trained model was then trained using the resized image. Table 2 contains the default settings for the training parameters of the proposed model.

To increase accuracy, these parameters are modified, including the optimizer, learning rate, and epochs. Figure 3 depicts the proposed pre-trained model's training plot. This model's training recognition accuracy improves to 94.35% once these parameters are adjusted. Table 3 represents the performance metrics of a FASTER RCNN that is designed to detect and classify bikes using the wheel rim and headlamp of a motorcycle.

Average Precision (AP), measures the accuracy of the model in terms of its ability to correctly identify the presence of a bike, as well as its ability to correctly identify the specific type or model of the bike.

The value of 0.9435 indicates a relatively high level of accuracy. Average Recall (AR), indicates the proportion of actual bike instances that were correctly identified by the model.

The value of 0.738 suggests that while the model is accurate, there is room for improvement in terms of correctly identifying all instances of

Table 4. Labels in the confusion matrix

1	super splendor -headlamp	28	hero pleasure - wheel
2	super splendor wheel	29	Ducati Panigale v4
3	fzs headlamp	30	Panigale headlamp
4	fzs wheel	31	access headlamp
5	Jawa 42 headlamp	32	access wheel
6	Jawa 42 wheel	33	honda shine headlamp
7	gixxer sf headlamp	34	honda shine wheel
8	gixxer sf wheel	35	apache rtr headlamp
9	tork Kratos headlamp	36	apache rtr wheel
10	tork Kratos	37	ktm duke headlamp
11	r15 headlamp	38	ktm duke wheel
12	r15 wheel	39	tvs sport headlamp
13	ktm adventure headlamp	40	tvs sport wheel
14	ktm adventure wheel	41	honda activa headlamp
15	mt15 headlamp	42	honda activa wheel
16	mt15 wheel	43	jawa yezdi headlamp
17	apache rr headlamp	44	jawa yezdi wheel
18	apache rr wheel	45	honda unicorn headlamp
19	bmw G310R headlamp	46	honda unicorn wheel
20	bmw G310R wheel	47	pulsar rs 200 headlamp
21	ktm rc headlamp	48	pulsar rs 200
22	ktm rc wheel	49	pulsar ns 200 headlamp
23	hero splendor plus-headlamp	50	pulsar ns 200
24	hero splendor plus-wheel	51	Jupiter headlamp
25	kawasaki ninja headlamp	52	Jupiter wheel
26	kawasaki ninja wheel	53	ntorq headlamp
27	hero pleasure - headlamp	54	ntorq wheel

bikes in the dataset. Localization loss measures the error in the model's ability to accurately localize the object in the image. The value of 0.065122 indicates that the model is performing well in this regard.

The classification loss measures the error in the model's ability to accurately classify the type or model of the bike. The value of 0.167798 suggests that the model is performing reasonably well in this regard, but there is room for improvement. Regularization loss measures the degree of regularization in the model, which is designed to prevent overfitting.

The value of 0.000000 suggests that the model does not overfit the training data. Total_loss is the sum of the three previous loss values and represents the overall error of the model.

The value of 0.275017 indicates that the model is performing well, but there is still room for improvement in terms of correctly identifying all instances of bikes in the dataset, as well as accurately classifying the type or model of the bike.

Figure 5 shows a decreasing trend in the loss as the number of steps increases, with some fluctuations due to noise in the data or variations in the optimization process.

At the beginning of training, the localization loss is high and the model's predictions are poor, as indicated by the high point at the left side of the graph. However, as the training progresses, the loss decreases steadily, with occasional fluctuations, until it reaches a minimum.

Figure 6 shows the total loss decreases over the course of training, as the model becomes better at making predictions.

The graph shows some fluctuations in the loss due to noise in the data or variations in the optimization process.

At the beginning of training, the loss is high because the model's predictions are poor.

Table 5. Performance analysis of proposed model with other models

Deep Learning Techniques	Accuracy
Mobile Det	89.5%
Yolo V5	90%
Proposed Network (Faster Rcnn)	94.35%
Mobile Det	89.5%

However, as the training progresses, the loss decreases steadily, with occasional fluctuations, until it reaches a minimum.

In Figure 7, the learning rate schedule for the training process is illustrated. Initially, a high learning rate, such as 0.04, is set to allow the optimization algorithm to quickly search the parameter space and find a good initial solution.

During the course of training, the learning rate gradually decreased based on a pre-defined schedule. In the given example, the learning rate is reduced by a factor of 10 after every 1000 steps.

Therefore, after 2000 steps, the learning rate is reduced to 0.03, and after 3000 steps, it is further reduced to 0.02, and so on. The proposed work's performance can be evaluated using the confusion matrix, as shown in figure 8 And the labels of each model are shown in table 4.

Precision, recall, and the F1 score are used to evaluate the proposed model performance. There are 27 classes, each class has two labels hence 54 subclasses in these 54 subclasses in 54 subclasses 48 subclasses have accuracy above 90%. The proposed model outperforms the others since 94.35% of the Bike models were correctly classified in the testing phase.

Table 5 shows the accuracy of each model on the task, as a percentage. MobileDET achieved an accuracy of 89.5%, YOLO v5 achieved an accuracy of 90%, and the proposed network using Faster R-CNN achieved the highest accuracy of94.35%.

5 Conclusion

Detecting and classifying motorbikes on public roads is a challenging task, but it has become increasingly important due to the rising number of bike-related crimes and accidents. The proposed work focuses on identifying the bike model using the bike's wheel rim and headlamp, as these components are often visible even if the number plate is damaged or not present.

The proposed work consists of bike model database creation, training the bike model by Fater RCNN, and testing the bike model identification.

This work achieved better identification accuracy of 94.35%. The proposed Fater RCNN model compared with YOLO v5 and MobileDET. Ultimately, this work has the potential to make our roads safer and help law enforcement agencies in their efforts to prevent and solve crimes.

References

1. **Ozkok, F. O. (2017).** A new approach to determine Eps parameter of DBSCAN algorithm. International Journal of Intelligent Systems and Applications in Engineering, Vol. 4, No. 5, pp. 247–251. DOI: 10.18201/ijisae.2017533899.
2. **Ng, J. Y., Tay, Y. H. (2012).** Image-based vehicle classification system. Proceedings of the 11th Asian-Pacific ITS Forum and Exhibition, pp. 1–11. DOI: 10.48550/ARXIV.1204.2114.
3. **Zehang, S., Miller, R., Bebis, G., Dimeo, D. (2002).** A real-time precrash vehicle detection system. Proceedings of the 6th IEEE Workshop on Applications of Computer Vision, pp. 171–176. DOI: 10.1109/acv.2002.1182177.
4. **Tan, J. K., Ishikawa, S., Sonoda, S., Miyoshi, M., Morie, T. (1970).** Moving objects segmentation at a traffic junction from vehicular vision. ECTI Transactions on Computer and Information Technology, Vol. 5, No. 2, pp. 73–88. DOI: 10.37936/ecti-cit.201152.54239.
5. **Shobha-Rani, B. R., Suparna, B. M., Teja, K. S. (2015).** Classification of vehicles using image processing techniques. International Journal of Engineering Research and Technology, Vol. 3, No. 21, pp. 1–4.
6. **Sarikan, S. S., Ozbayoglu, A. M., Zilci, O. (2017).** Automated vehicle classification with

image processing and computational intelligence. Procedia Computer Science, Vol. 114, pp. 515–522. DOI: 10.1016/j.procs.2017.09.022.

7. **Giron, N. N. F., Billones, R. K., Fillone, A., Del-Rosario, J. R., Cabatuan, M., Bandala, A., Dadios, E. P. (2020).** Motorcycle rider helmet detection for riding safety and compliance using convolutional neural networks. IEEE 12th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management, pp. 1–6. DOI: 10.1109/hnicem51456.2020.940 0149.
8. **Yuan, Y., Zhang, J., Wang, Q. (2018).** Bike-person re-identification: A benchmark and a comprehensive evaluation. IEEE Access, Vol. 6, pp. 56059–56068. DOI: 10.1109/access.2018.2872804.
9. **Sun, Z., Bebis, G., Miller, R. (2002).** On-road vehicle detection using Gabor filters and support vector machines. Proceedings of the 14th International Conference on Digital Signal Processing, Vol. 2, pp. 1019–1022. DOI: 10.1109/ICDSP.2002.1028263.
10. **Chiverton, J. (2012).** Helmet presence classification with motorcycle detection and tracking. IET Intelligent Transport Systems, Vol. 6, No. 3, pp. 259. DOI: 10.1049/iet-its.2011.0138.
11. **Tabassum, S., Ullah, S., Al-Nur, N. H., Shatabda, S. (2020).** Poribohon-BD: Bangladeshi local vehicle image dataset with annotation for classification. Data in Brief, Vol. 33, pp. 106465. DOI: 10.1016/j.dib.2020.106465.
12. **Espinosa, J., Velastin, S., Branch, J. (2018).** Motorcycle detection and classification in urban scenarios using a model based on faster r-CNN. Proceedings of the 9th International Conference on Pattern Recognition Systems, pp. 91–96 DOI: 10.1049/cp.2018.1292.
13. **Regenwetter, L., Curry, B., Ahmed, F. (2021).** Biked: a dataset for computational bicycle design with machine learning benchmarks. Journal of Mechanical Design, Vol. 144, No. 3, pp. 1–19. DOI: 10.1115/1.4052585.
14. **Figueiredo, A., Brayan, J., Reis, R. O., Prates, R., Schwartz, W. R. (2021).** MoRe: A large-scale motorcycle re-identification dataset. IEEE Winter Conference on Applications of Computer Vision, pp. 4033–4042. DOI: 10.1109/wacv48630.2021.0040 8.
15. **Wang, H., Hu, Z., Guo, Y., Yang, Z., Zhou, F., Xu, P. (2020).** A real-time safety helmet wearing detection approach based on CSYOLOv3. Applied Sciences, Vol. 10, No. 19, pp. 6732. DOI: 10.3390/app10196732.
16. **Kocamaz, M. K., Gong, J., Pires, B. R. (2016).** Vision-based counting of pedestrians and cyclists. IEEE Winter Conference on Applications of Computer Vision, pp. 1–8. DOI: 10.1109/wacv.2016.7477685.

Article received on 09/03/2023; accepted on 30/03/2023.
**Corresponding author is Lokesh V S.*

ISSN 2007-9737

Young Adults' Instagram Posts and Depressive Moods: A Study in Mexico in the Wild

Iván A. Encinas-Monroy[1], Jessica Beltrán[2], Luis H. Sánchez[3], Luis Felipe-Rodríguez[1], Adrián Macías[1], Cynthia B. Pérez[4], Manuel Domitsu[1], Luis A. Castro[1,*]

[1] Instituto Tecnológico de Sonora, Departamento de Computación y Diseño, Mexico

[2] Universidad Autónoma de Coahuila, Centro de Investigación en Matemáticas Aplicada, Mexico

[3] Instituto Politécnico Nacional, Centro de Investigación y Desarrollo de Tecnología Digital, Mexico

[4] Instituto Tecnológico de Sonora, Unidad Guaymas, Mexico

ivanalexx@msn.com, jessicabeltran@uadec.edu.mx, lh.sg26@gmail.com, {luis.rodriguez, adrian.macias, cynthia.perez, manuel.domitsu}@itson.edu.mx, luis.castro@acm.org

Abstract. Patterns of use of social networking sites like Instagram can be indicators of the mental state of users. Of particular interest to the HCI community are those markers and patterns useful for inferring the mental health of users experiencing depressive episodes or moods. Detecting individuals' depressive moods through their typical Instagram activity remains a challenge due to the diversity of the content posted. Previous research often focuses on retrieving content of hashtags related directly to depression for analysis. Thus, although based on real posts, results can be highly biased. Analyzing all user posts in individuals' day-to-day lives can yield ecologically valid findings, but it is challenging. We conducted an observational study aimed at detecting the depressive moods of users from their Instagram posts. We analyzed text, images, and posting behavior using two approaches: inferential statistics and machine learning. Our results indicate that the time of day and the hue levels of a posted image could lead to the detection of depressive moods. Furthermore, our machine-learning approach yielded up to 65% of accuracy. Although our study yields ecologically valid findings, several challenges remain to be addressed due to the heterogeneity of the dataset, as it typically happens in real-world studies.

Keywords. Social networking sites, depressive mood detection, Instagram, machine learning, behavior analysis, image analysis, text analysis, transfer learning.

1 Introduction

Depression is one of the leading causes of disability around the world, affecting about 300 million people [42]. Detecting and treating depression in young people is therefore paramount. Young people tend to seek help and support on social networking sites (SNS) through externalization [16, 2], where they share how they feel about certain situations or topics.

In these situations, their friends and relatives can help them by showing support or care [3].

Instagram posts describing antidepressant use have increased exponentially from 2010 to 2018 [14]. Instagram acknowledges the importance of this topic by showing a message when a user searches for images related to certain keywords such as depressed (see Figure 1).

Users have reported five primary social and psychological motives for using Instagram: social interaction, archiving, self-expression, escapism, and peeking [22]. Images and text shared in posts can be analyzed to unravel patterns that can signal the presence of depression, such as the preferences of colors [6], certain topics through images or captions [2, 3], the use of certain words [38], image filters [31], or explicitly expressing depressive symptoms [20].

Social media markers have been reported as a valid way to detect depression [20, 31]. On Instagram, these markers significantly differ between depressed people and nondepressed people, which include the number of Instagram followers, frequency of Instagram use or content of messages, and filters [20, 31].

A qualitative analysis of depression-related posts on Instagram revealed different types of disclosures individuals make in the context of depression-tagged posts [3]. In [17], authors found that people with depressive symptoms are more likely to prefer the use of Twitter over Instagram and Facebook.

Also, in [24] was reported that more frequent Instagram use had associations with greater depressive symptoms when users reported a high proportion of strangers followed. Finally, negative social comparison has been one of the reasons for depressive symptoms [24, 26].

With regards to Facebook, having few Facebook friends and mutual friends, posting frequently, and using few location tags are positively correlated with depressive symptoms [20]. Nonetheless, other works report that Facebook can work as a protective factor against depressive symptoms [17].

In this work, we focus on the in-the-wild usage of Instagram to study depressive moods since it is an SNS that has been popular with younger generations.

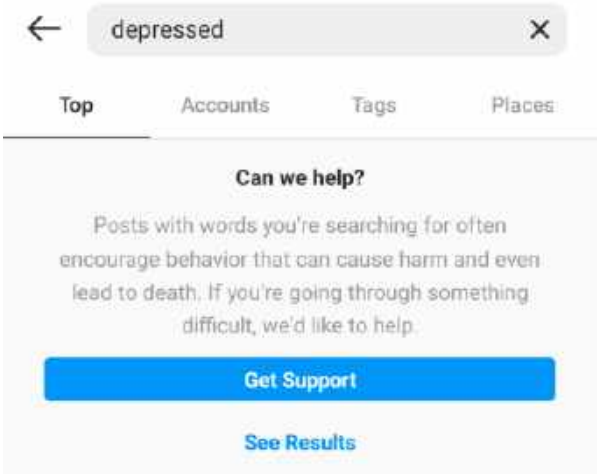


Fig. 1. Screenshot of the Instagram mobile app when looking for images related with keyword depressed

In addition, Instagram's nature of promoting oneself and telling others about what is happening during your day contrasts with other discussion-oriented SNSs (e.g., Twitter) [28], which makes Instagram a suitable platform to analyze users' content and investigate whether it can reveal their moods.

As opposed to previous works that use surveys [24, 23, 22, 17, 26, 30, 33] or analyze posts filtered by depression-related hashtags such as #depression or #depressed [2, 3, 14, 29, 1], in this work we developed a tool to collect Instagram posts of young adults (i.e., our participants) and ask them to answer the PANAS-X inventory each time they posted.

Next, we analyzed those images, text, and posting behavior to associate them with depressive moods as users go by in their daily lives. We used two approaches: inferential statistics and machine learning.

Analyzing in-the-wild posts can be challenging since there may be posts that are completely unrelated to the posters' moods. In this regard, our study yields ecologically valid results and understandings of how SNS are used in real life [4, 7].

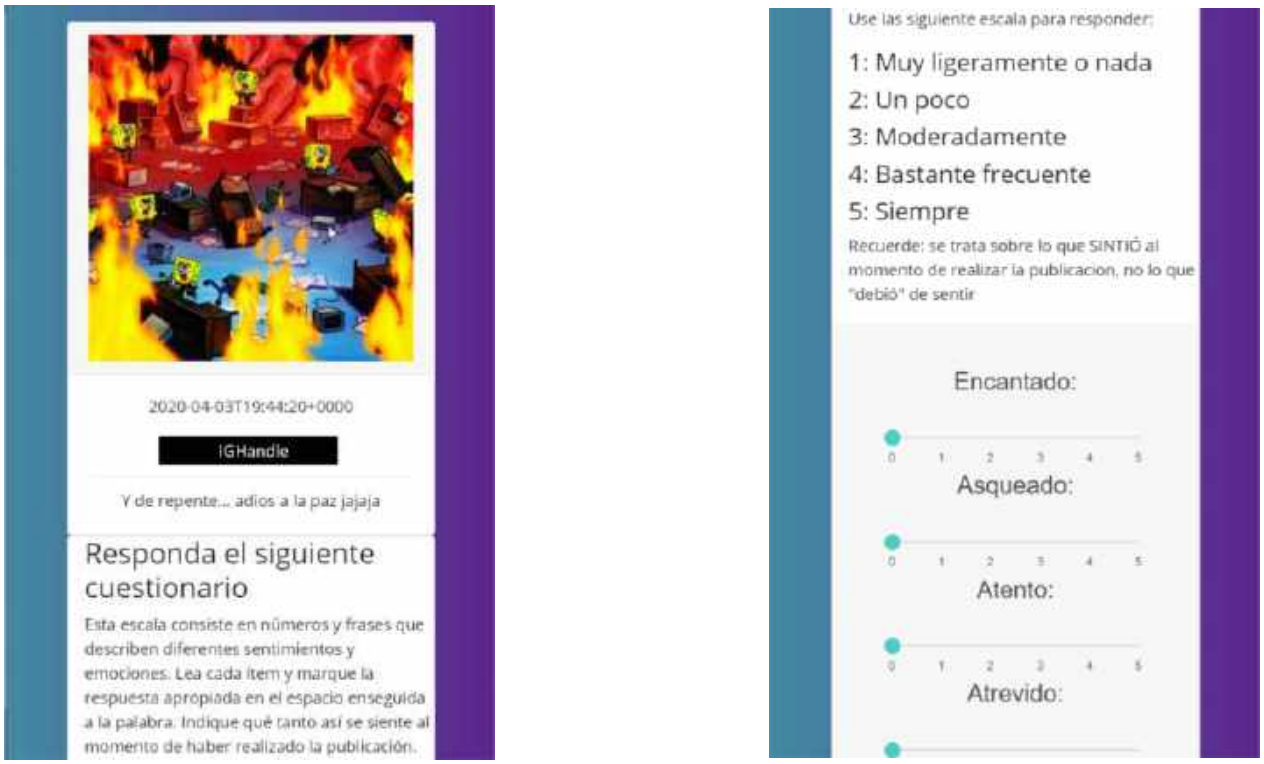


Fig. 2. Snapshots of the web-based app (In spanish)

2 Related Work

The interest in identifying users' internal states through Instagram posts has increased in the last decade [24, 23, 22, 17, 26, 30, 33, 2, 3, 14, 29]. Observational studies typically use self-report data to find associations between psychological inventories and behavior [24, 23, 22, 17, 26, 30, 33]. Other works, however, have focused on analyzing Instagram posts seeking behavior patterns from images, text, and emoticons. In this section, we describe works that focus on the analysis of text, images, and posting behavior in SNS.

2.1 Text Analysis in SNS

Text analysis tools and techniques have been increasingly used to get insights into the users' internal states (e.g., mood and emotions), or other psychological traits [38, 18]. Text analysis has been particularly used to explore and predict different mental disorders through posts on SNSs [9, 11].

Previous works have shown that certain keywords in SNS posts can be used to identify individuals with mental disorders. For instance, these types of posts include hashtags such as #depression, #anxiety or #suicide, among other words that may be related to mental disorders such as the names of antidepressants [14] or the name of the disorder itself [9, 10, 27].

For instance, in [10] it was proposed a lexicon of depressed users on Twitter and found that some recurrent themes were related to symptoms (e.g., anxiety, withdrawal, severe, delusions), disclosure (e.g., fun, play, helped, god), treatment (e.g., medication, side-effects, doctor, doses), and relationships (e.g., home, woman, she, him). Authors from [34] showed that college students with depression often use more personal singular pronouns at the moment of writing. Although

Fig. 3. Random sample of 50% of the images posted by individuals who were categorized in the minimal, mild, moderate, and severe classes. Images have been intentionally blurred for privacy

this study was carried out with written essays, these findings could potentially be extrapolated to SNS posts.

Another approach often used is sentiment analysis since keywords and words used by users with mental disorders are often charged with negative emotions [11, 43]. For example, it has been reported that some words such as issues, bad, or anxiety could be used to predict the jump from depression to suicidal thoughts [11].

Some of these words had high frequency like other less negative ones such as make, around, time, when, where, and others. However, this could raise some concerns as people might not be completely honest on SNS by purposely undermining their own negative feelings so others do not feel bad for them [12] or by expressing more positive emotions than they are actually experiencing since it might attract more attention to the post [37].

Text analysis per se can be challenging, but it can be more difficult if analysis is carried out without further context. For a more precise interpretation of the users' moods, more data associated with the users moods at the time of posting is required.

2.2 Image Analysis on SNS

The content of shared images can include data about the user's interests and potentially about their mental state or mood. Examples of these

Table 1. Face data extracted by Google cloud vision API grouped by Beck's class

Beck's Class	Users (N)	Images (N)	Faces (N)	Faces (Max N)	Faces (Mean)	Positive Emotion (Mean)	Negative Emotion (Mean)
Minimal	17	151	138	10	0.9139	25.0066	14.7615
Mild	5	65	33	10	0.5076	18.0461	12.1538
Moderate	7	53	67	8	1.2641	27.0377	20.4905
Severe	6	56	55	10	0.9821	18.6428	21.6250

Table 2. Face data extracted by Google cloud vision API grouped by the PANAS-X main categories

PANAS-X Category	Images (N)	Faces (N)	Faces (Max N)	Faces (Mean)	Positive Emotion (Mean)	Negative Emotion (Mean)
Positive	180	184	10	1.0222	28.2722	13.5888
Negative	45	35	8	0.7777	15.3777	29.0222
Neutral Positive	44	32	6	0.7272	20.4772	20.6818
Neutral Negative	56	42	6	0.7500	13.2857	11.6785

types of content include the number of faces (i.e., individuals) in the image, the predominant color, and the types of objects, among others.

The content of Instagram images has been analyzed and linked to different aspects of users. For example, when studying the relationship between personality traits and gender, and the images posted, researchers found a link between extraversion and gender of users [21].

Likewise, [36] extracted objects using Microsoft Azure Cognitive Services with the aim of classifying images into thousands of categories, such as car, city, interior, and others, and used them to determine the age and gender of users. The relationship between mental health and shared photos has also been studied.

For example, in [27] studied the relationship between several visual attributes of the images such as color, themes or emotions, and self-disclosures of Instagram users related to their mental health. Images shared on SNSs have also been used for detecting depression.

In [31] analyzed the content of images and data such as the number of posts per day or the number of likes to compare nondepressed and depressed individuals. Features such as the number of faces in the photos and color properties were related to depression.

However, the dataset entries were observed and labeled through crowdsourcing with Amazon's Mechanical Turk, which could have introduced a bias since emotions can be interpreted differently by third parties.

Moreover, the user who posted the image might have a salient emotion at the moment of posting, which could have been missed by annotators since they are only looking at the image.

Then, having third parties annotate the images of others can be a challenge to derive adequate findings, since different cultures or experiences can shape or bias the annotations.

In another study, [46] used a deep regression network (deemed DepressNet) to analyze faces since they might indicate a depressive disorder. However, apart from selfies, users on Instagram typically post different types of images, such as landscapes, artwork, or pets, which makes it difficult to deploy in the wild.

Finally, in [8] used multimodal data from Instagram posts, including the content of images, text, and user's behavior, to detect users with depressive moods. For the image analysis,

Table 3. Number of images with animals and sketches grouped by the Beck's class

Beck's Category	Animals	Sketches	% of Total Images by Beck's Category
Minimal	14	16	19.87
Mild	10	12	33.85
Moderate	4	2	11.32
Severe	9	2	19.64
Total	37	32	84.68

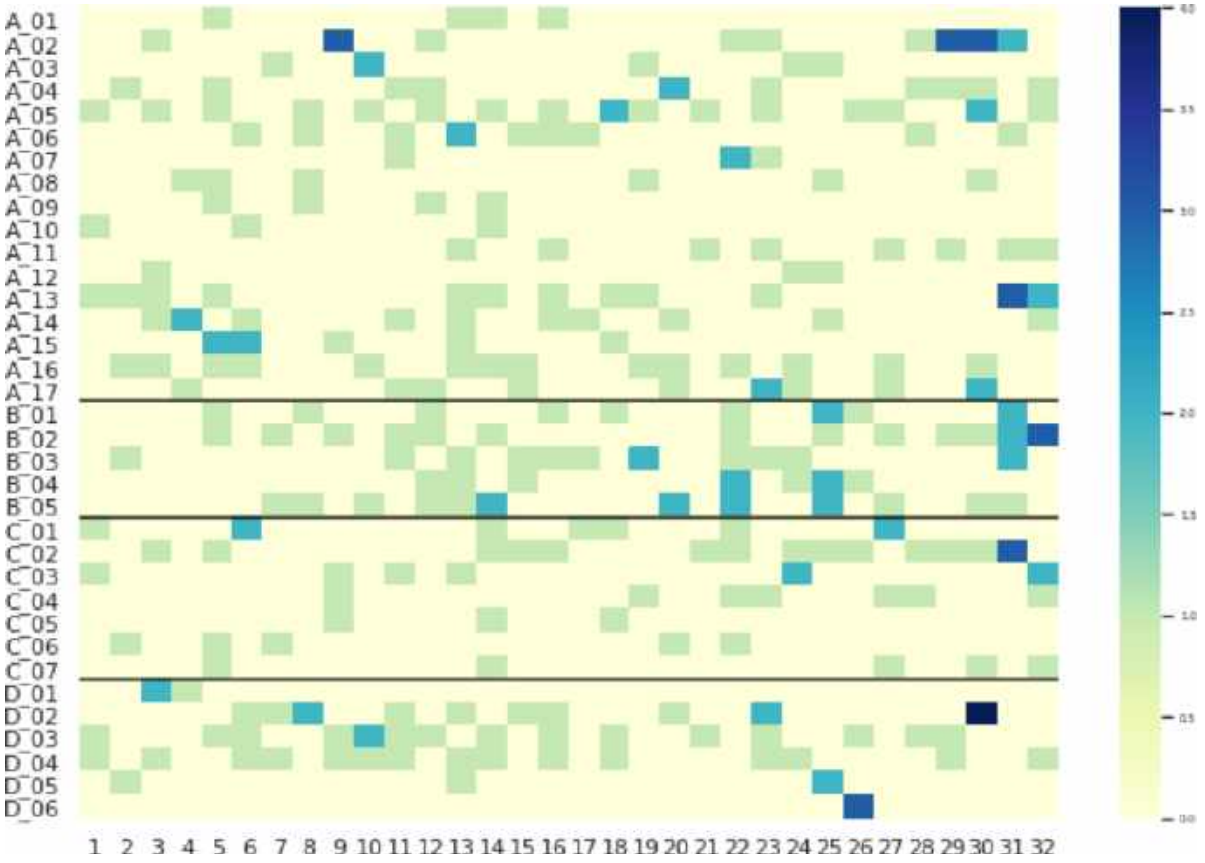


Fig. 4. Posting activity from all users throughout the duration of the study (x-axis = days; y-axis = participants)

used the AlexNet Convolutional Neural Network for transfer learning to get a prediction score of depressive images.

Afterward, they merged individual predictions for image, text, and behavior for an overall prediction. Although the dataset was obtained from real users, they retrieved the dataset by searching specific depression keywords on Instagram users' profiles, which could have biased the results toward individuals who self-describe as depressed.

In general, there is a need for conducting studies that analyze data coming from individuals with their typical behavior on SNS, i.e., ecologically valid findings. Moreover, research must consider a

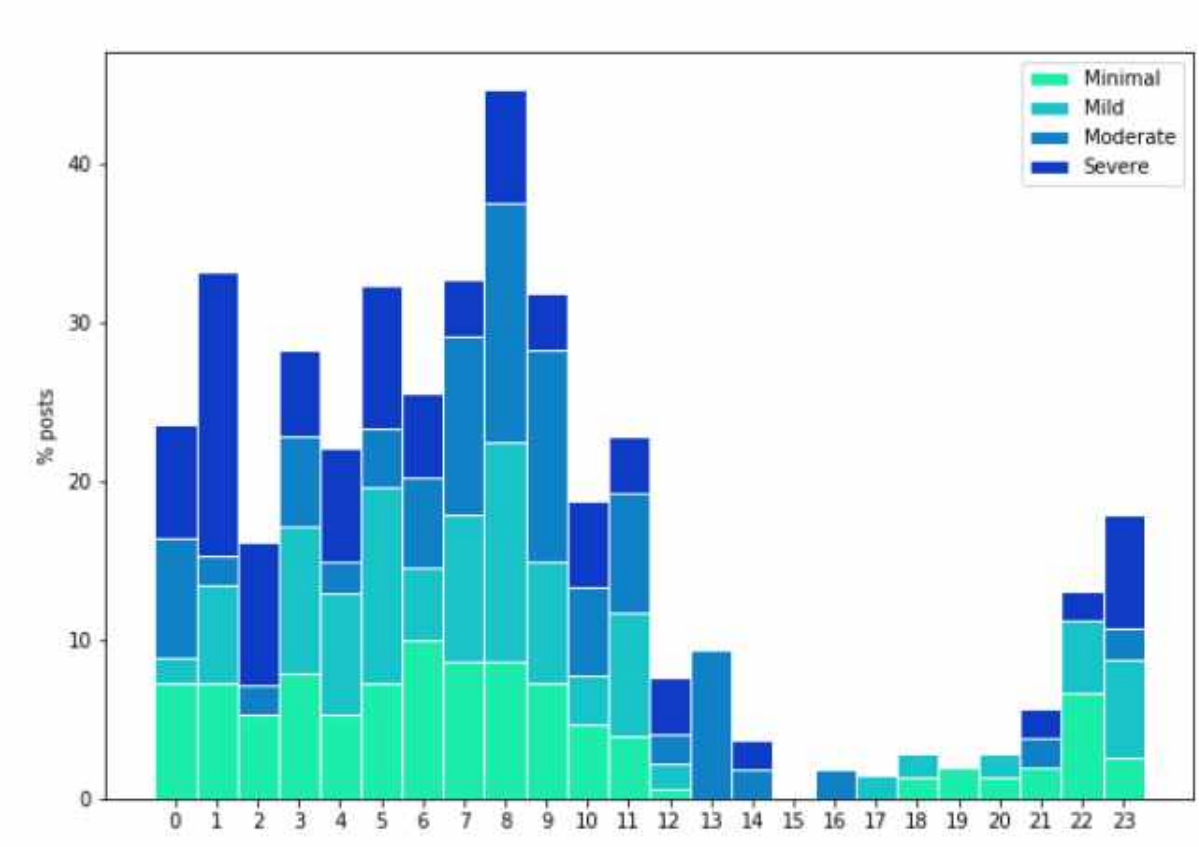


Fig. 5. Posts binned per the hour of day (x axis = time of day; y axis = number of posts)

wider range of posts from users [29] as opposed to selecting posts with specific hashtags (e.g., #depression), which can bias the results and our understanding of the manner in which these aspects take place in the real world.

3 Methods

We carried out an observational study to collect users' Instagram posts from which we analyzed text, images, and posting behavior (i.e., time of post).

In this section, we describe the participants, research procedure, dataset, and data preprocessing.

3.1 Participants

We used a convenience sampling method to recruit participants. The invitations to participate in the study were sent through electronic media such as WhatsApp or Facebook Messenger.

We recruited 50 individuals from Northwest Mexico, from which 35 participants (13 male) remained until the end of the study.

Our participants were, on average, 23.51 years old (SD = 3.36), ranging 19-40 years old. Sixteen (16) of them (45.71%) were university students.

34% of the participants said they were regular users of Instagram. All our participants were native speakers of Spanish. All participants signed an informed consent. No monetary incentive was given.

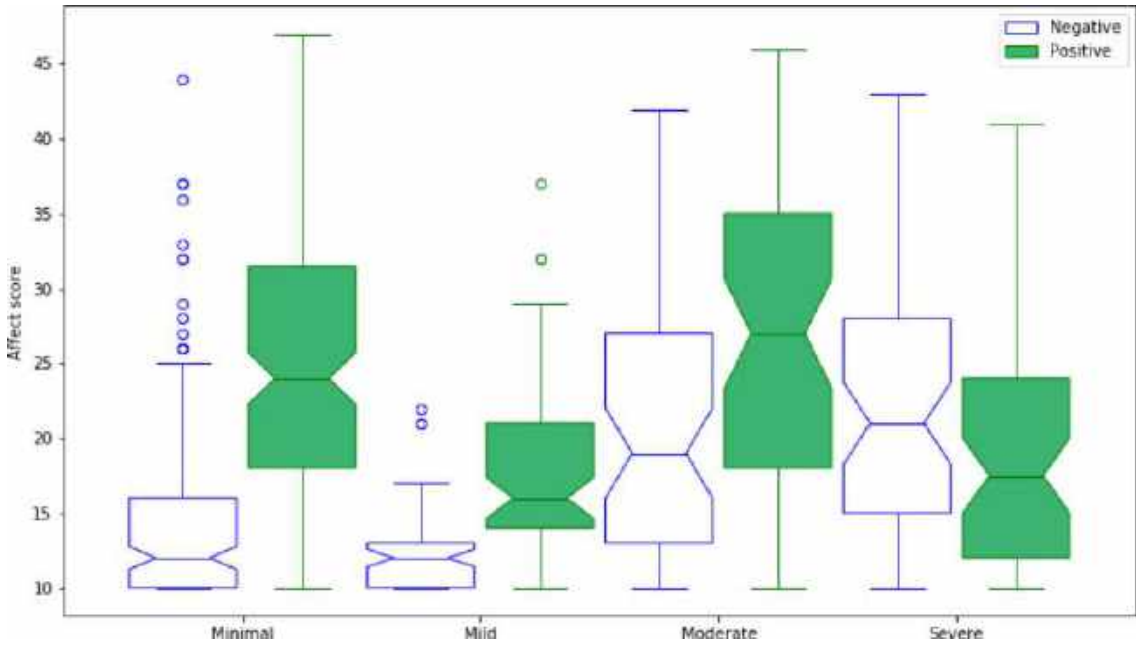


Fig. 6. General negative affect (GNA) and general positive affect (GPA) from PANAS-X across the Beck's classes

3.2 Instruments

We used the following instruments to obtain data in the wild.

3.2.1 PANAS-X

We used the validated Spanish version of the PANAS-X [32], which has 46 items to measure the positive affect and negative affect using a 5-point Likert-like scale (1 = Lightly or nothing; 5 = Always).

The original English version consists of 60 items [41]. The PANAS-X shows two different kinds of categories for both ends of the valence spectrum: General Positive Affect (GPA) and General Negative Affect (GNA), as well as the Basic Positive Affect (BPA) and Basic Negative Affect (BNA). GPA and GNA are directly related to the results of the more commonly used PANAS [41], as they are composed of the same items.

Their basic counterparts are composed of different kinds of items that are only present in PANAS-X and reach other kinds of emotions like fear, sadness, guilt, hostility, joviality, self-assurance, and attentiveness.

The rationale behind PANAS-X's positive and negative affect is that people are able to feel both kinds of emotions at the same time in the same high or low intensity levels.

It is possible to detect intense joy or happiness while also detecting strong feelings of sadness or anger through these questionnaires. Such cases can be related to being confused about what one can feel about certain situations.

3.2.2 Beck's Depression Inventory

Beck's depression inventory (BDI) [35] is a Spanish version of a psychological test used to evaluate the depressive symptomatology of people.

The BDI has 21 items designed to assess the severity of symptoms of depression in adults and adolescents. The BDI has a score range of 0 to 63, depending on the option selected by the person.

This score helps researchers and health professionals categorize the level of depression according to people's symptoms in order to identify its intensity or evaluate its therapeutic progress.

Table 4. Pearson's R correlation between PANAS-X affect scores, and SentiStrength and Google AutoML

		SentiStrength Positive Affect	SentiStrength Negative Affect	Google AutoML Natural Language Value
GPA	r	0.037	-0.004	0.101
	N	325.0	325.0	325.0
	p	0.256	0.474	**0.035**
GNA	r	-0.069	-0.086	-0.108
	N	325.0	325.0	325.0
	p	0.106	0.061	**0.025**

Table 5. Contingency table showing the number of posts per week

Beck's Class	Week 1	Week 2	Week 3	Week 4	Week 5	Total
Minimal	32	36	27	28	28	**151**
Mild	5	16	10	21	13	**65**
Moderate	10	9	8	16	10	**53**
Severe	13	16	8	12	7	**56**
Total	**60**	**77**	**53**	**77**	**58**	**325**

3.2.3 Web-Based App for Data Collection

We developed a web-based app (Figure 2) to retrieve the users' latest posts using their Instagram (IG) handle. Every time the users posted on IG, our web-based app retrieved the posted image, text, and date.

Below the retrieved image from the IG post, the app displayed the PANAS-X questionnaire so that our participants could rate the types of emotions felt at the moment of posting that particular image.

This strategy differs from previous approaches since it provides not only the data linked to the post (e.g., image, text, timestamp) but also about their emotions at the time of posting.

3.3 Data Collection Protocol

Thirty-five individuals participated in this study who received a demographic questionnaire and Beck's depression inventory (BDI) in Spanish [35], which they responded to online without supervision.

First, we explained to the participants the general purpose of the study. Also, we asked them about their frequency of use of IG and IG handle. Of the 35 participants, 17 obtained a Beck score that falls in the category of minimal depression (score 0-13), 5 in mild depression (score 14-19), 7 as moderate depression (score 20-28), and 6 with severe depression (score 29 or more). The procedure was as follows:

- For 32 days, participants had to use Instagram as they would typically use it. We suggested our participants post 4 times per week, although this was not compulsory.
- After each post, they were asked to answer the PANAS-X that corresponded to that publication using our Web-based app.
- In the event that the participant posted and did not answer the corresponding PANAS-X after a few hours, one of the authors sent a reminder via the WhatsApp or Instagram messaging service.

4 Data Preprocessing and Feature Extraction

A total of 325 entries were posted by the 35 participants. On average, each user posted 9.28 times (SD = 4.94) throughout the duration of the study. From the 325 images, 151 were posted by participants in the category of minimal depression,

Table 6. Contingency table showing the number of posts per time of day

Beck's class	Morning	Afternoon	Night	Total
Minimal	85	5	61	**151**
Mild	44	2	19	**65**
Moderate	35	7	11	**53**
Severe	27	1	28	**56**
Total	**191**	**15**	**119**	**325**

Table 7. Results with the 16-day time window

Classifier	Accuracy	Precision	Recall	F1-Score
SVM	0.65	0.65	0.55	0.55
RF	0.62	0.49	0.66	0.46
LR	0.65	0.65	0.55	0.55

according to Beck's depression inventory, 65 by participants in the mild class, 53 by those in the moderate class, and 56 by participants who were categorized as severely depressed. Figure 3 shows a random sample of the images posted by users from all depression categories.

From the 325 entries, 46 entries consisted of image-only publications. From these, 18 entries came from participants categorized in the minimal class, 19 from participants in the mild class, 2 from participants in the moderate class, and 7 from those classified as severely depressed, according to Beck's depression inventory.

From the 279 posts that included text, they included a mean text description of 11.59 words (SD = 19.91). Each text record consisted of the following: post ID (Integer), IG handle (string), timestamp (Integer), text description (string), image URL (string), and type of post (string: carousel, video, or image).

In summary, the dataset consisted of 325 image files, 279 text records, and 325 46-tuple vectors, i.e., one PANAS-X answer per post. To analyze the content of posts, we extracted features by preprocessing the data using state-of-the-art tools.

4.1 Text Processing

For studying the link between the text of an IG post and the emotions reported by the participants, we used two different tools that identify the general emotion from the given text and also provide additional information that can be related to depression, such as the amount of singular or plural pronouns, which has been reported to be relevant in depressed students [34].

The first tool was the Google AutoML Natural Language [13], which delivers the magnitude and value score of the identified emotion in the text, pronoun count, first-person pronouns, first-person singular pronouns, plural pronouns, and first-person plural pronouns.

The second tool used was the Spanish version of SentiStrength [40], original version by [39], which delivers the negative and positive scores from the identified emotion in the text. In the case of SentiStrength, we removed emoticons since the tool is unable to detect them and can only interfere the analysis.

Finally, we also computed the number of characters and words, the ratio of the number of pronouns over the number of words, the ratio of first-person pronouns over total pronouns, and the ratio of plural pronouns over total pronouns. In total, we obtained 14 features from the text.

4.2 Image Processing

To extract features from the image dataset, we used the state-of-the-art Automated Machine Learning (AutoML) Vision by Google Cloud Platform[1], which is an implementation of AutoML for image classification and object detection.

It consists of an Application Programming Interface (API) that offers machine learning models that assign labels and detects objects in images, and it can also be used to train personalized models of machine learning. In total, we obtained 9 features from images. Color perception has been suggested as a marker of mood [5], where grayer and darker colors are related to depressive moods.

We included the dominant Red (R), Green (G), and Blue (B) colors of each photo provided by the Google Cloud service. Also, the levels of hue and saturation have been of interest to researchers as they could possibly indicate levels of sadness or depression [31].

[1] cloud.google.com/

Table 8. Results with the 7-day (week) time window

Classifier	Accuracy	Precision	Recall	F1-score
SVM	0.62	0.61	0.53	0.55
RF	0.51	0.47	0.43	0.43
LR	0.59	0.58	0.59	0.58

Table 9. IG dataset content

Class	Training	Validation
Predominantly depressive	138 images	35 images
Barely depressive	121 images	30 images

Drawing on this, the mean values of Hue, Saturation, and Value (HSV) were retrieved from the images through the scikit-image library[2].

We obtained relevant object labels from images that can be used to relate to a depressive mood. For example, the label face, since according to []reece2017instagram, the number of faces reflects a greater social interaction, which can be related to less tendency to depression. Table 1 and Table 2 show data related to faces and their emotions as detected by Google Cloud Vision API.

Since the dataset features images with humans, pets, landscapes, drawings or anything the user felt like posting, we also obtained labels animal, if there were animals in the photo, and sketch, since we identified that several images were drawings, cartoons or sketches. Table 3 shows the data with this type of content.

4.3 Behavior Data Processing

Mental status can be related to posting behavior, such as the time or frequency of posting activity. In Figure 4, we show a heatmap with the posting activity from all our participants.

A darker color indicates more publications during each day. For the purposes of this figure, we grouped the users by Beck's class in the Y axis (A_xx = minimal depression, B_xx = mild, C_xx = moderate, D_xx = severe).

Figure 5 shows the time of day in which participants posted the most, grouped by the participants' severity of depression.

[2]scikit-image.org/

It can be seen that there is a sharp increase of activity from 10 PM to 5 AM in our participants, particularly those categorized as severely depressed. Most posts occurred from midnight to noon across all groups. From behavior processing, we obtained 3 features: date, time, and time between consecutive posts in minutes.

5 Results

In this section, we present the results of analyzing data collected from the observational study to detect depressive moods of users from their Instagram posts using two approaches:

1. Inferential statistics.
2. Machine learning techniques.

5.1 Inferential Statistics Approach

5.1.1 Relating the Severity of Depression and the Emotions Linked to a Post

Detecting depressive moods in IG posts where there are no clear signals of sadness, such as gloomy pictures or certain keywords, can be challenging. For this, we first need to identify the types of emotions related to a particular post, i.e., the emotions experienced by the participants at the moment of posting.

For this, we used t-student tests to explore the relationship between the level of depression as approximated by the Becks' inventory (administered at the beginning of the study), and the general positive/negative emotions through the PANAS-X (administered each time the user posted).

As shown in Figure 6, for the General Positive Affect (GPA), participants with depressive moods (i.e., mild, moderate, and severe) had lower scores ($\mathrm{GPA} = 20.98$, $\mathrm{SD} = 8.84$) when compared with users with minimal depressive moods ($\mathrm{GPA} = 25.00$, $\mathrm{SD} = 9.22$), which was statistically significant ($t = 4.01$, $\mathrm{df} = 312$, $P < 0.0001$).

As for the General Negative Affect (GNA), participants with depressive moods had a significantly higher score ($\mathrm{GNA} = 17.74$, $\mathrm{SD} = 6.51$) when compared with users with

Table 10. MART dataset content across the two classes

Class	Paintings
Negative	183 images
Positive	317 images

minimal depressive moods ($\mathrm{GNA} = 14.76$, $\mathrm{SD} = 8.09$), which was statistically significant ($t = -3.67$, $\mathrm{df} = 321, P < 0.0001$). Also, participants with severe depression had a mean GPA of 18.64 ($\mathrm{SD} = 7.30$), while the rest of the classes had an average GPA score of 23.72 ($\mathrm{SD} = 9.35$) ($t = -4.49$, $\mathrm{df} = 97$, $P < 0.0001$).

On the other hand, participants with severe depression also reported higher scores of the GNA ($\mathrm{mean} = 21.62$; $\mathrm{SD} = 8.41$) when compared with the average GNA ($\mathrm{mean} = 15.26$; $\mathrm{SD} = 6.87$) of the rest of the participants ($t = 5.3$, $\mathrm{df} = 71$, $P < 0.0001$). This suggests that the PANAS-X helps discriminate negative and positive emotions associated with a particular post. Even more, these feelings seem to concur with Beck's classes.

5.1.2 Text Analysis

We obtained the Pearson's R correlation between the reported PANAS-X's GPAs and GNAs and the scores from the SentiStrength and Google AutoML sentiment analysis tools (Table 4). As shown in Table 4, there is a positive correlation between the GPA and the Google AutoML Natural Language Value ($r = 0.101$, $N = 325$, $p = 0.035$).

Also, there is a negative correlation between Google AutoML Natural Language Value and the GNA ($r = -0.108$, $N = 325$, $p = 0.025$), contrary to what was expected. Although weak, these correlations show that there is a relationship between the GPA and the GNA and the valence detected by Google AutoML and that there is an association between the affect related to a particular post and the affect of the text.

Also, SentiStrength's results were not statistically significant, but it is interesting that with both tools, the GNA was slightly more correlated with the text than the GPA, which could potentially suggest that people are more expressive about their negative feelings rather than their positive ones. Still, this is inconclusive.

5.1.3 Image Analysis

One of our interests is to better understand certain markers that describe depressive moods in Instagram posts. [31] reported that people with depression are more interested in "bluer" o "darker blue" images in comparison with nondepressed. Since hue levels are directly related to the blue color of an image.

We conducted an analysis of images posted by our participants. The hue levels from the images posted by participants with depressive moods ($\mathrm{mean} = 0.40$, $\mathrm{SD} = 0.09$) were higher than those with minimal depressive moods ($\mathrm{mean} = 0.37$, $\mathrm{SD} = 0.09$).

The mean difference was 0.03, which was statistically significant ($t = -2.13$, $\mathrm{df} = 316$, $p = 0.017$). This means that participants with depression generally posted images that were slightly "bluer". Also, there were fewer faces in the photos posted by participants with depressive moods ($\mathrm{mean} = 0.89$, $\mathrm{SD} = 1.71$) than in those by nondepressed ($\mathrm{mean} = 0.91$, $\mathrm{SD} = 1.44$). The mean difference was 0.02, which was not significant ($t = -0.13$, $\mathrm{df} = 323$, $p = 0.45$).

5.1.4 Behavior Analysis

As shown in Figure 4, there are slight differences between the groups. The group in the mild class seems to have more activity than the rest of the groups. A chi-square test was used to explore the association between the severity of depression and the number of posts per week. Table 5 shows the contingency table. There was no significant association between the severity of depression of the participant and the number of posts per week ($X^2 = 13.35$, $\mathrm{df} = 12$, $p = 0.34$).

Previous works have reported signals of possible posting patterns in depressed individuals [10]. That is, depressed individuals tend to be more active at night than non-depressed ones. For this analysis, we binned posts into three times of day (morning: 4:00 AM - 12:59 PM, afternoon: 1:00 PM - 7:59 PM, night: 8:00 PM - 3:59 AM). The chi-square test was used to explore the association between the severity of depression (i.e., Beck's class) and the number of posts per time of the day.

Table 11. IG and MART: Merged dataset

Class	Training	Validation
Predominantly Depressive	285 images	71 images
Barely Depressive	373 images	94 images

Following Table 6, there was a significant difference between Beck's class and the time of day at the moment of posting ($X^2 = 20.85$, $\mathrm{df} = 6$, $p < 0.01$). Therefore, the evidence collected suggests that posting time is associated with Beck's class in which our participants were categorized.

5.2 A Classic Machine Learning Approach

We used a classic machine learning approach to classify the participants' Beck's class based solely on the data shared in their posts. For this, we represented each participant with a vector that we used for training models. Afterward, we evaluated the models on unseen vectors. Since we have a small dataset, we used N-fold cross-validation for the evaluation.

5.2.1 Target Classes

We used binary target classes. For this, we combined mild, moderate, and severe Beck's classes into a single class deemed as Predominantly depressive and the other remaining class minimal deemed as Barely depressive.

This arrangement categorized 17 subjects within the class Barely depressive and 18 within the class Predominantly depressive. However, we removed the data from participants A_01, A_07, A_09, A_10, A_12, C_05, D_01, D_05, and D_06 since they posted infrequently and spent about two weeks without posting, making feature vectors invalid in all these cases.

This exclusion leads to 12 subjects in Barely depressive class and 14 in Predominantly depressive class.

5.2.2 Feature Vectors

We preprocessed each post and obtained the 26 features described in Section 4 such as magnitude and value of text emotion, count of pronouns, number of faces in the image, time of posting, and others. Since a single post is not enough to characterize a person with depression, we aggregate data from several posts to obtain a 52-tuple vector composed of the Mean and Standard Deviation (SD) per feature.

We run experiments with two different time windows to aggregate features and construct the vectors. The first time window comprised 16 days, which produced 50 vectors. Lastly, a 7-day time window resulted in 116 vectors. We performed a correlation analysis across the vector elements, and dropped those highly correlated.

For instance, the SD of the number of characters had a 99% correlation with the SD of the number of words, so we only kept one of those. The correlation threshold to drop features was set to 90%, thus the feature vector size was reduced to 37.

5.2.3 Binary Classification

We used the normalized 37-feature vectors to train machine learning models using the Python Scikit-learn library[3]. We run experiments with the following algorithms: Support Vector Machines (SVM), Random Forest (RF), and Logistic Regression (LR) using 10-fold cross-validation. We run a hyperparameter grid search in all algorithms, and we report the best results.

The hyperparameter search space was {C: 0.1, 1, 10, 30, 40, 50; Gamma: 0.5, 1, 5, 10} for SVM, {C: 0.001, 0.1, 1, 10, 30, 40, 100; Penalty: "l1", "l2"; Solver: "liblinear", "saga"} for LR, and {emax depth: 30, 50, 60, 80, 100; minimum samples per leaf: 2, 3, 5, 8; minimum samples per split: 4, 7, 8, 10; number of estimators: 60, 80, 90, 100, 150} for RF.

Additionally, we evaluated using different top k features that resulted from the feature selection process during cross-validation. We next compared the results using Accuracy, Precision,

[3] scikit-learn.org/

Table 12. IG and Art photo: Merged dataset

Class	Training	Validation
Predominantly Depressive	459 images	142 images
Barely Depressive	404 images	125 images

Recall, and F1-score evaluation metrics per 16 days (Table 7) and 7-day (Table 8). It can be seen that SVM and LR outperform RF. In the case of a 7-day time frame, SVM had higher Accuracy and Precision but lower Recall than LR.

The best features for the 16-day window (Table 7) were text Google value score average, word count in first person voice average, animal flags average, value of the image, red intensity average, green intensity, and blue intensity average.

The best hyperparameters for the 16-day window were C=30 and $\gamma = 0.001$ for SVM; max depth=40, max features=1, and n estimators=80 for RF; and C=0.01, penalty=12, and solver= libnear for LR. In addition, the best features for the 7-day window (Table 8) were the text Google value score average, word count in first person voice average, pronouns ratio average, animal flags average, and value of the image.

The best hyperparameters for the 7-day window were: C=10 and $\gamma = 5$ for SVM; max depth=20, max features=4 and n estimators=100 for RF; and C=1, penalty=11 and solver= libnear for LR. We can see that features that produce better results are similar across all classifiers. In fact, the features “animal flags average” and “value” remain relevant across all classifiers.

5.3 A Deep Learning Approach

We also utilized deep learning (DL) algorithms to learn about the potential to discriminate between depressed and nondepressed individuals based on images. We used the same aforementioned classes: Predominantly depressive and Barely depressive. Due to a small dataset, we used transfer learning, which is a DL approach where learned features in a pre-trained model with larger datasets are transferred to a second network with other target tasks and data.

The transfer learning approach has proven to be powerful when there are data in a target network [45]. The base model used for the transfer learning was pre-trained with the ImageNet dataset[4], which contains more than 14 million images within 1, categories.

The architecture used is ResNet50 [15], from which the weights of the pre-trained model were transferred to our target classification task, removing the output layer of ResNet50 and adding the connected layers of classification by image. We used the library Tensorflow[5] for the experiments. We performed four different experiments in this DL approach.

For the 4 experiments, we used transfer learning with the model with ResNet50 architecture. The key difference among the 4 experiments was the datasets used. Since we have a small dataset, we sought to augment it with similar datasets labeled from people with depression or labeled as positive or negative images.

As mentioned, our participants shared images with artistic content and drawings besides photos of people and landmarks. We found datasets with these types of content that are labeled as positive and negative, which we next describe.

In total, we used 3 datasets for these four experiments (E1, E2, E3, E4), merging some of them for the experiments: a) the dataset we collected (i.e., IG dataset); b) a dataset from abstract paintings from the MART museum from Italy (i.e., MART dataset); c) and a dataset from artistic images showing emotions (i.e., Art photo dataset). We next list the experiments:

– E1: ResNet50 with IG
– E2: ResNet50 with IG + MART
– E3: ResNet50 with IG + Art photo
– E4: ResNet50 with IG + MART + Art photo

During these experiments, the training across the 4 experiments was made through 45 epochs. We tested with dropout regularization to avoid overfitting.

[4] www.image-net.org/
[5] www.tensorflow.org/

Table 13. IG, MART, and Art Photo: Merged dataset

Class	Training	Validation
Predominantly Depressive	597 images	188 images
Barely Depressive	641 images	203 images

In the following sections, we describe each of the following experiments with mode details. The results of those will be shown in the following pages.

5.3.1 E1: ResNet50 with IG

As mentioned, we used 2 classes: Predominantly depressive and Barely depressive. For E1, the IG dataset was split into training data (80%) and validation data (20%). The data distribution was done as shown in Table 9.

5.3.2 E2: ResNet50 with IG + MART

In [44] applied a methodology to classify 500 abstract paintings from the MART museum from Italy. They used 2 classes: Positive perception and Negative perception.

They performed a statistical classification in which 100 participants reported their first impression of the abstract painting on a 1 – 7 scale (1 = Negative perception; 7 = Positive perception).

Authors defined those with average scores lower or equal to 4 as the negative class and those with average scores above 4 as the positive. The distribution of classes is shown in Table 10.

In E2, we merged the IG dataset with the dataset created by [44], where the samples from the Negative class were merged with the samples from the Predominantly depressive class.

Similarly, the samples in the Positive class were merged with the samples of the Barely depressive class. Table 11 shows the resulting dataset split into two: training (80%) and validation (20%).

5.3.3 E3: ResNet50 with IG + Art Photo

Authors from [25] proposed a methodology to extract emotional features of images, like color, composition, hue, and others, with the aim of classifying images using the presented emotion. They used a dataset with 806 artistic images.

These images were obtained through an image hosting website using the search terms: Amusement, Awe, Contentment, Excitement as positive emotions, and Anger, Disgust, Fear, Sad to represent negative emotions.

The images were created by artists who sought to awaken a specific emotion to the viewer through the color manipulation, illumination, composition, and so forth.

Like the previous example, we merged these images with the IG dataset, taking the images with negative class as Predominantly depressive and the images with positive class as Barely depressive. The total of images with configuration of 80% for training and 20% for validation is shown in Table 12.

5.3.4 E4: ResNet50 with IG + MART + Art Photo

The last dataset used is a combination of the three datasets previously described. This resulted in a new dataset with images close to those an average user can post on Instagram (i.e., real, artistic, or abstract images). The total of images with a configuration of 80% training and 20% validation is shown in Table 13.

5.3.5 Classification Results with DL

Figure 7 and Figure 8 show the training and validation accuracy from the four experiments. As we can see, the results are lower than 0.60 accuracy across all the experiments in both the training and validation sets. This experiment provides an understanding of the potential of using the algorithms in this problem using all the data for training and validation.

For further exploration, we trained and validated using only the MART dataset, not including the Instagram data. The accuracy results are 1 in training and 0.75 in validation, showing overfitting but, most importantly, showing that adding the

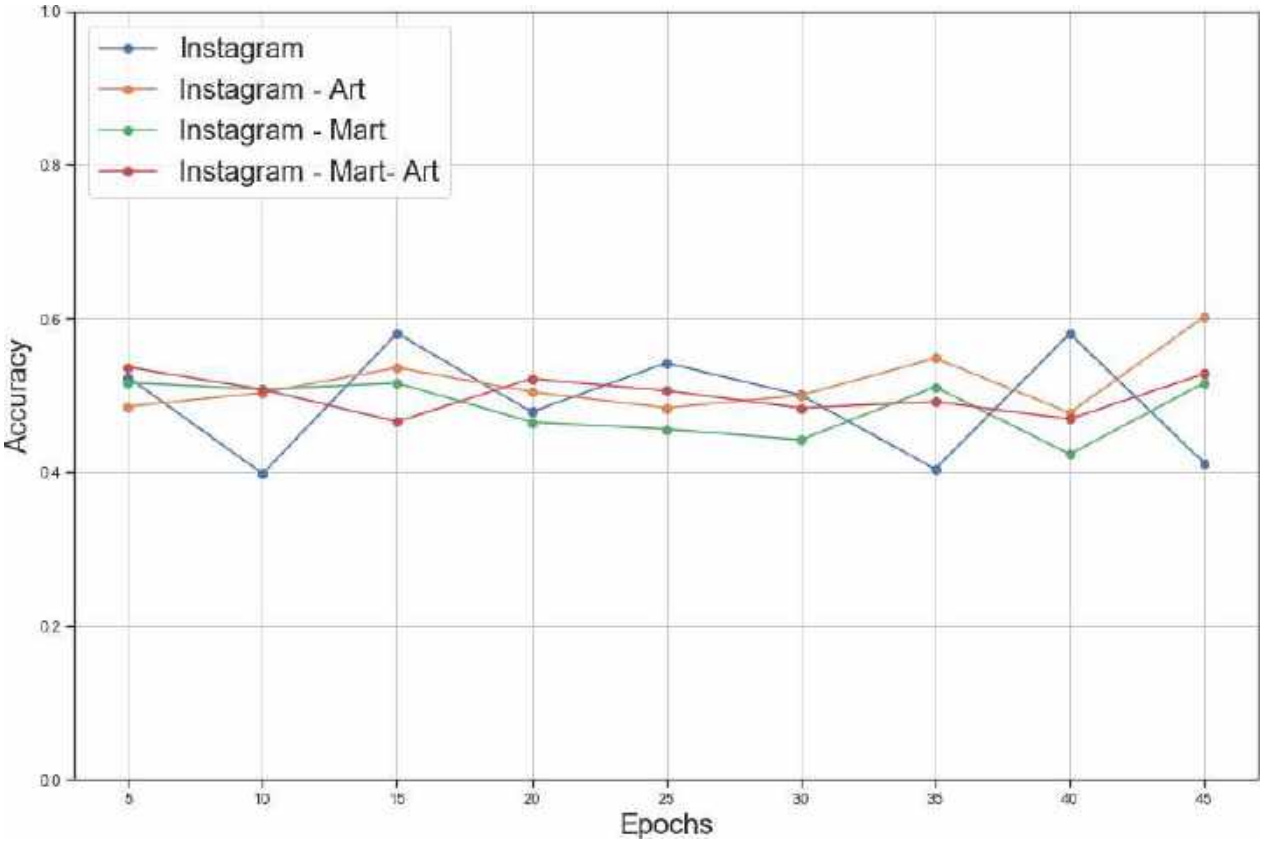


Fig. 7. Training accuracy through 45 epochs

Instagram datasets significantly decreased results. This can be caused due to different reasons, such as the image distributions from both datasets being different or the labels positive or negative not aligning well with the classes from the Beck's depression inventory.

In general, just like most available datasets, both the MART dataset and the Art photo dataset were labeled a posteriori by third parties and do not properly reflect the emotions when a photo is taken or posted by an individual.

On the contrary, our Instagram dataset was collected in the wild, where our participants were categorized according to the Becks' depression inventory, who might be privately dealing with their condition, showing subtle signs through their posts while not being truly explicit about it. The results obtained through deep learning are not conclusive.

We believe that these results can be improved by combining more information from the posts, such as the image captions. However, we wanted to show these results, which can illustrate the difficulty of this problem. More studies of this kind are needed to deepen the understanding of this problem and the way the DL can help tackle it.

6 Discussion

There are several aspects that can be of interest to the HCI community, and that can be used in real-world applications for monitoring patients who have been diagnosed with certain disorders, such as depressive disorders. First, from our results, we can see that the PANAS-X can help identify emotions related to a particular post, having found that those with depressive moods have a higher

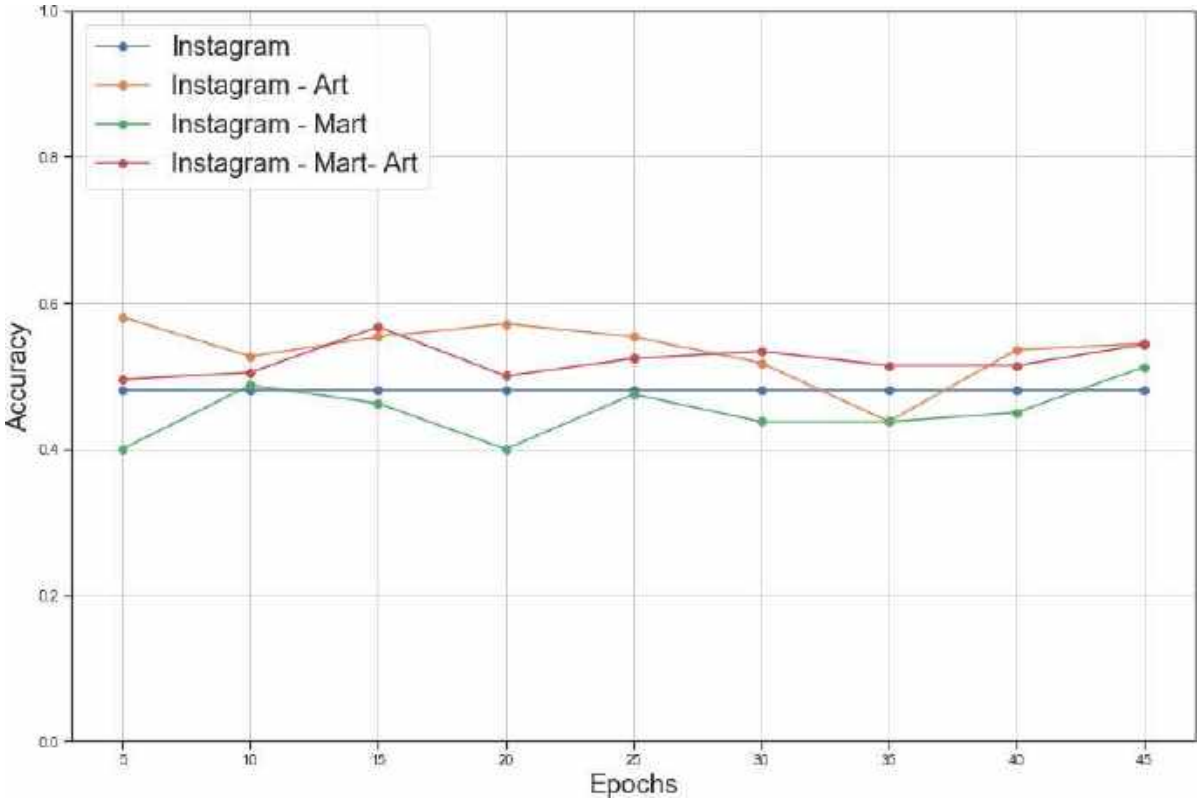


Fig. 8. Validation accuracy through 45 epochs

GNA and a lower GPA than those who barely have depressive moods. Also, as expected, those with prominent depressive moods (i.e., severe depression) have a higher GNA and lower GPA when compared with all other Beck's classes.

Even when this aspect was asked every time a participant posted an image on Instagram, these emotions seemed to be generalized as they spanned across the entire set of participants, i.e., there was an actual difference between the groups.

As for the images published, we found that participants who have prominent depressive moods generally posted images that are bluer, that is, the prominent underlying color is blue.

This concurs with other studies that report that blue or grayscale colors are preferred by depressed individuals [6]. The results from our study help us understand that these findings can also be replicated in Mexican young adults (i.e., our participants).

Although this is not conclusive due to statistical power, our results suggest that this is an interesting avenue to explore. Also, from the posting behavior, the number of posts by our participants (i.e., posting behavior) did not vary across all the study between groups.

The time of day, however, seems to be a good discriminant between those with prominent depressive behaviors and those who were classified as minimally depressed.

The obtained results with machine learning are higher than chance, which proves the feasibility of using these types of algorithms to be potentially used to detect depression with reservations.

However, better ways to discriminate the types of posts are paramount for designing and training appropriate machine learning models. Possibly, for this particular problem, we may need multi-modal classification from data coming from various sources, rather than focusing only on IG data.

ISSN 2007-9737

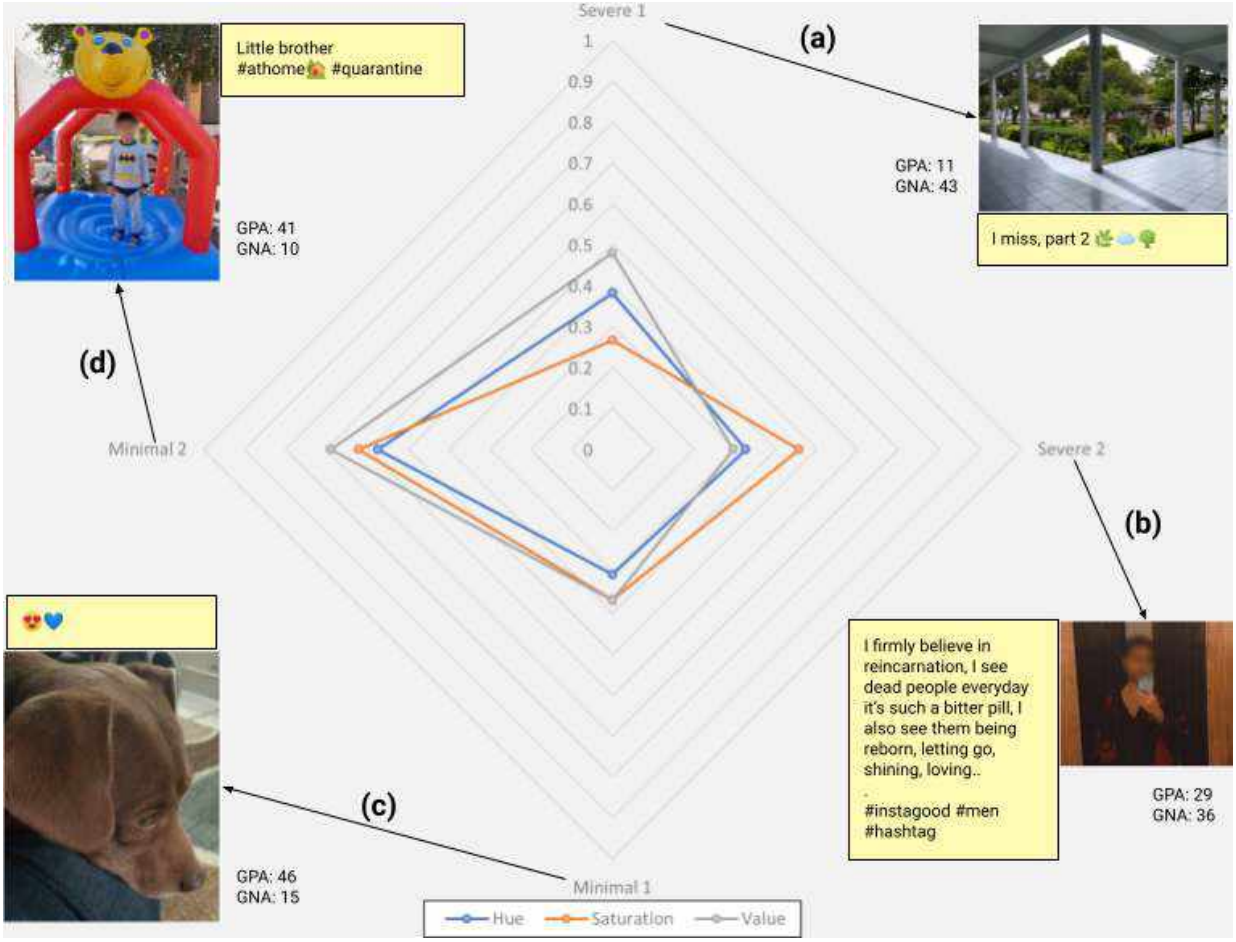


Fig. 9. Posts from four participants: (a) Severe 1: GPA=11, GNA=43; (b) Severe 2: GPA=29, GNA=36; (c) Minimal 1: GPA=46, GNA=15; (d) Minimal 2: GPA=41, GNA=10. The maximum value for GPA and GNA is 50. Faces have been blurred for privacy

Still, our results are valuable in that is one of the first studies trying to understand whether in-the-wild behaviors such as the use of IG can be used to infer certain behaviors of interest.

6.1 Ecological Validity

The in-the-wild nature of this study can be double-edged. On the one side, the findings derived from the data can be ecologically valid [4, 7], since the dataset we collected is similar to what young adults are posting on Instagram in their day-to-day lives.

Using hashtagged posts can cause filter bias during analysis [29]. To avoid this, in this work, we considered all types of posts by users during a four-week window, which can be difficult for algorithms to discriminate due to the noise derived from heterogeneous content published by users.

One of the challenges in this work is the diversity in the data across various levels of depression, as categorized by Beck's depression inventory. For instance, an image shared by a person with severe depression might be similar in terms of colors or lighting to an image shared by a person without any particular sign of depression.

At the same time, people with the same severity of depression might share unrelated or opposite images. To illustrate the heterogeneity of the dataset, Figure 9 shows the HSV levels and GPA/GNA data related to four posts from different users who were categorized either as minimal or severe, according to Beck's depression inventory.

We also show the actual images, and the caption associated, which have been translated to English by the authors. Following, Figure 9a and Figure 9b, we can see that Hue and Value (ie., brightness) are similar, but not in terms of Saturation. Also, both GPA are relatively low (GPAa=11, GPAb=29) and both GNA are high (GNAa=43, GNAb=36).

We want to highlight that both images were posted by participants categorized as severely depressed. Interestingly, these images yielded similar values to Figure 9c in terms of hue, saturation, and value (HSV). In other words, images Figure 9a and Figure 9b look alike to their polar opposite, Figure 9c.

Presumably, one expected type of image with high GNA and low GPA from our severely depressed participants is Figure 9c, which is dark and essentially colorless. This image strongly differs from Figure 9d, even when both have similar values of GPA (GPAc=46; GPAd=41) and GNA (GNAc=15, GNAd=10). Taking all of these aspects into account, using machine learning approaches for classifying these sorts of images is not trivial.

In most related work, the challenge associated with using complete datasets from the day-to-day lives of users has not been accounted for since datasets generally originate from homogeneous groups (e.g., online communities for depression support) or commonly used hashtags on SNS (e.g., #depression).

Since posts can have similar characteristics stated from the moment that they are posted, much more homogeneous data within target classes can be obtained, and higher variations between classes can have a significant positive impact on the performance of machine learning approaches. As we have seen in this work, having a more diverse dataset has a negative impact on the performance, especially due to the size of the dataset.

6.2 Limitations and Scope

One limitation of the present study is the quantity of data collected. For the statistical and machine learning approaches, it is highly beneficial to have large datasets (in this case, posts and users) to be able to yield better results. Therefore, this study could have benefited from a larger sample of users over a longer period.

7 Conclusion

We presented an analysis of posts collected in the wild from users of Instagram. From our results, we can conclude that identifying depressive moods from Instagram posts can be challenging since participants typically post about their inner states as much as about their interests, preferences, hobbies, or even memes.

Images do not necessarily relate to feelings or emotions, but may also be associated with situations and interpersonal strategies (e.g., social status). The results of this work can be summarized as follows: some behaviors can be potentially used to discriminate depressed from nondepressed users, such as the time of posting and the hue color of the images.

The in-the-wild nature of this study yielded ecologically valid results, but further user context could be useful for adequate results in classification. As seen when merged with other datasets, our dataset noisy.

Future work includes collecting more data from more participants and over a longer period, performing experiments with different combinations of information gathered from posts such as text, images, and perhaps additional context such as filters or location.

Also, we suggest creating subcategories of images such as people, locations, pets, or other predefined categories, which could help increase classification performance, but with additional overhead.

Acknowledgments

We thank our participants for their valuable time and effort. This work was partially funded with a scholarship provided to the first and third authors, and the Instituto Tecnológico de Sonora through the PROFAPI program.

References

1. **Ahmed, A., Aziz, S., Toro, C. T., Alzubaidi, M., Irshaidat, S., Serhan, H. A., Abd-Alrazaq, A. A., Househ, M. (2022).** Machine learning models to detect anxiety and depression through social media: A scoping review. Computer Methods and Programs in Biomedicine Update, Vol. 2, pp. 100066. DOI: 10.1016/j.cmpbup.2022.100066.
2. **Andalibi, N., Ozturk, P., Forte, A. (2015).** Depression-related imagery on instagram. Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative Work and Social Computing, pp. 231–234. DOI: 10.1145/2685553.2699014.
3. **Andalibi, N., Ozturk, P., Forte, A. (2017).** Sensitive self-disclosures, responses, and social support on instagram: The case of #depression. Proceedings of the ACM Conference on Computer Supported Cooperative Work and Social Computing, pp. 1485–1500. DOI: 10.1145/2998181.2998243.
4. **Andrade, C. (2018).** Internal, external, and ecological validity in research design, conduct, and evaluation. Indian Journal of Psychological Medicine, Vol. 40, No. 5, pp. 498–499. DOI: 10.4103/ijpsym.ijpsym_334_18.
5. **Barrick, C. B., Taylor, D., Correa, E. I. (2002).** Color sensitivity and mood disorders: biology or metaphor?. Journal of affective disorders, Vol. 68, No. 1, pp. 67–71. DOI: 10.1016/S0165-0327(00)00358-X.
6. **Carruthers, H. R., Morris, J., Tarrier, N., Whorwell, P. J. (2010).** The Manchester color wheel: Development of a novel way of identifying color choice and its validation in healthy, anxious and depressed individuals. BMC Medical Research Methodology, Vol. 10, No. 1. DOI: 10.1186/1471-2288-10-12.
7. **Carter, S., Mankoff, J., Klemmer, S. R., Matthews, T. (2008).** Exiting the cleanroom: On ecological validity and ubiquitous computing. Human–Computer Interaction, Vol. 23, No. 1, pp. 47–99. DOI: 10.1080/07370020701851086.
8. **Chiu, C. Y., Lane, H. Y., Koh, J. L., Chen, A. L. P. (2020).** Multimodal depression detection on instagram considering time interval of posts. Journal of Intelligent Information Systems, Vol. 56, No. 1, pp. 25–47. DOI: 10.1007/s10844-020-00599-5.
9. **Coppersmith, G., Harman, C., Dredze, M. (2014).** Measuring post traumatic stress disorder in twitter. Proceedings of the International AAAI Conference on Web and Social Media, Vol. 8, No. 1, pp. 579–582. DOI: 10.1609/icwsm.v8i1.14574.
10. **De-Choudhury, M., Gamon, M., Counts, S., Horvitz, E. (2013).** Predicting depression via social media. Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media, Vol. 36, No. 1–2, pp. 168–169. DOI: 10.3109/01460862.2013.798190.
11. **De-Choudhury, M., Kiciman, E., Dredze, M., Coppersmith, G., Kumar, M. (2016).** Discovering shifts to suicidal ideation from mental health content in social media. Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 2098–2110. DOI: 10.1145/2858036.2858207.
12. **Dibble, J. L., Levine, T. R. (2013).** Sharing good and bad news with friends and strangers: Reasons for and communication behaviors associated with the mum effect. Communication

Studies, Vol. 64, No. 4, pp. 431–452. DOI: 10.1080/10510974.2013.770407.

13. **Google (2017).** AutoML for large scale image classification and object detection. ai.googleblog.com/2017/11/automl-for-large-scale-image.html.
14. **Gupta, R., Ariefdjohan, M. (2020).** Mental illness on instagram: a mixed method study to characterize public content, sentiments, and trends of antidepressant use. Journal of Mental Health, Vol. 30, No. 4, pp. 518–525. DOI: 10.1080/09638237.2020.1755021.
15. **He, K., Zhang, X., Ren, S., Sun, J. (2016).** Deep residual learning for image recognition. IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778. DOI: 10.1109/cvpr.2016.90.
16. **Hunt, J., Eisenberg, D. (2010).** Mental health problems and help-seeking behavior among college students. Journal of Adolescent Health, Vol. 46, No. 1, pp. 3–10. DOI: 10.1016/j.jadohealth.2009.08.008.
17. **Jeri-Yabar, A., Sanchez-Carbonel, A., Tito, K., Ramirez-del-Castillo, J., Torres-Alcantara, A., Denegri, D., Carreazo, Y. (2018).** Association between social media use (twitter, instagram, facebook) and depressive symptoms: Are twitter users at higher risk?. International Journal of Social Psychiatry, Vol. 65, No. 1, pp. 14–19. DOI: 10.1177/0020764018814270.
18. **Jianqiang, Z., Xiaolin, G. (2017).** Comparison research on text pre-processing methods on twitter sentiment analysis. IEEE Access, Vol. 5, pp. 2870–2879. DOI: 10.1109/access.2017.2672677.
19. **Kiersz, A., Allan, A. (2019).** Suicide is gen Z's second-leading cause of death, and it's a worse epidemic than anything millennials faced at that age. www.businessinsider.com/cdc-teenage-gen-z-american-suicide-epidemic.
20. **Kim, J., Uddin, Z. A., Lee, Y., Nasri, F., Gill, H., Subramanieapillai, M., Lee, R., Udovica, A., Phan, L., Lui, L., Iacobucci, M., Mansur, R. B., Rosenblat, J. D., McIntyre, R. S. (2021).** A systematic review of the validity of screening depression through facebook, twitter, instagram, and snapchat. Journal of Affective Disorders, Vol. 286, pp. 360–369. DOI: 10.1016/j.jad.2020.08.091.
21. **Kim, Y., Kim, J. H. (2018).** Using computer vision techniques on instagram to link users' personalities and genders to the features of their photos: An exploratory study. Information Processing and Management, Vol. 54, No. 6, pp. 1101–1114. DOI: 10.1016/j.ipm.2018.07.005.
22. **Lee, E., Lee, J. A., Moon, J. H., Sung, Y. (2015).** Pictures speak louder than words: Motivations for using Instagram. Cyberpsychology, Behavior, and Social Networking, Vol. 18, No. 9, pp. 552—556. DOI: 10.1089/cyber.2015.0157.
23. **Lin, L. Y., Sidani, J. E., Shensa, A., Radovic, A., Miller, E., Colditz, J. B., Hoffman, B. L., Giles, L. M., Primack, B. A. (2016).** Association between social media use and depression among U.S. young adults. Depression and Anxiety, Vol. 33, No. 4, pp. 323–331. DOI: 10.1002/da.22466.
24. **Lup, K., Trub, L., Rosenthal, L. (2015).** Instagram #Instasad?: Exploring associations among instagram use, depressive symptoms, negative social comparison, and strangers followed. Cyberpsychology, Behavior, and Social Networking, Vol. 18, No. 5, pp. 247–252. DOI: 10.1089/cyber.2014.0560.
25. **Machajdik, J., Hanbury, A. (2010).** Affective image classification using features inspired by psychology and art theory. Proceedings of the 18th ACM International Conference on Multimedia, pp. 83–92. DOI: 10.1145/1873951.1873965.
26. **Mackson, S. B., Brochu, P. M., Schneider, B. A. (2019).** Instagram: Friend or foe? The application's association with psychological well-being. New Media and Society, Vol. 21, No. 10, pp. 2160–2182. DOI: 10.1177/1461444819840021.

27. **Manikonda, L., De-Choudhury, M. (2017).** Modeling and understanding visual attributes of mental health disclosures in social media. Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 170–181. DOI: 10.1145/3025453.3025932.

28. **Manikonda, L., Meduri, V. V., Kambhampati, S. (2021).** Tweeting the mind and instagramming the heart: Exploring differentiated content sharing on social media. Proceedings of the International AAAI Conference on Web and Social Media, Vol. 10, No. 1, pp. 639–642. DOI: 10.1609/icwsm.v10i1.14819.

29. **McCosker, A., Gerrard, Y. (2020).** Hashtagging depression on instagram: Towards a more inclusive mental health research methodology. New Media and Society, Vol. 23, No. 7, pp. 1899–1919. DOI: 10.1177/1461444820921349.

30. **Pittman, M., Reich, B. (2016).** Social media and loneliness: Why an instagram picture may be worth more than a thousand twitter words. Computers in Human Behavior, Vol. 62, pp. 155–167. DOI: 10.1016/j.chb.2016.03.084.

31. **Reece, A. G., Danforth, C. M. (2017).** Instagram photos reveal predictive markers of depression. EPJ Data Science, Vol. 6, No. 1. DOI: 10.1140/epjds/s13688-017-0110-z.

32. **Robles, R., Páez, F. (2003).** Estudio sobre la traducción al español y las propiedades psicométricas de las escalas de afecto positivo y negativo (panas). Salud Mental, Vol. 26, No. 1, pp. 69–75.

33. **Rozgonjuk, D., Pruunsild, P., Jürimäe, K., Schwarz, R. J., Aru, J. (2020).** Instagram use frequency is associated with problematic smartphone use, but not with depression and anxiety symptom severity. Mobile Media and Communication, Vol. 8, No. 3, pp. 400–418. DOI: 10.1177/2050157920910190.

34. **Rude, S., Gortner, E. M., Pennebaker, J. (2004).** Language use of depressed and depression-vulnerable college students. Cognition and Emotion, Vol. 18, No. 8, pp. 1121–1133. DOI: 10.1080/02699930441000030.

35. **Sanz, J., Perdigón, A. L., Vázquez, C. (2003).** Adaptación española del inventario para la depresión de Beck-II (BDI-II): 2. Propiedades psicométricas en población general. Clínica y Salud, Vol. 14, No. 3, pp. 249–280.

36. **Song, J., Han, K., Lee, D., Kim, S. W. (2018).** "Is a picture really worth a thousand words?": A case study on classifying user attributes on Instagram. PLoS One, Vol. 13, No. 10, pp. e0204938. DOI: 10.1371/journal.pone.0204938.

37. **Stieglitz, S., Dang-Xuan, L. (2013).** Emotions and information diffusion in social media—sentiment of microblogs and sharing behavior. Journal of Management Information Systems, Vol. 29, No. 4, pp. 217–248. DOI: 10.2753/mis0742-1222290408.

38. **Tausczik, Y. R., Pennebaker, J. W. (2009).** The psychological meaning of words: LIWC and computerized text analysis methods. Journal of Language and Social Psychology, Vol. 29, No. 1, pp. 24–54. DOI: 10.1177/0261927x09351676.

39. **Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., Kappas, A. (2010).** Sentiment strength detection in short informal text. Journal of the American Society for Information Science and Technology, Vol. 61, No. 12, pp. 2544–2558. DOI: 10.1002/asi.21416.

40. **Vilares, D., Thelwall, M., Alonso, M. A. (2015).** The megaphone of the people? Spanish SentiStrength for real-time analysis of political tweets. Journal of Information Science, Vol. 41, No. 6, pp. 799–813. DOI: 10.1177/0165551515598926.

41. **Watson, D., Clark, L. (1999).** The PANAS-X: Manual for the positive and negative affect schedule-expanded form. Psychology Publications. DOI: 10.17077/48vt-m4t2.

42. **World Health Organization (2021).** World mental health day. www.who.int/news-room/fact-sheets/detail/depression.

43. Yang, Y., Jia, J., Zhang, S., Wu, B., Chen, Q., Li, J., Xing, C., Tang, J. (2014). How do your friends on social media disclose your emotions?. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 28, No. 1. DOI: 10.1609/aaai.v28i1.8740.

44. Yanulevskaya, V., Uijlings, J., Bruni, E., Sartori, A., Zamboni, E., Bacci, F., Melcher, D., Sebe, N. (2012). In the eye of the beholder: Employing statistical analysis and eye tracking for analyzing abstract paintings. Proceedings of the 20th ACM International Conference on Multimedia, Association for Computing Machinery, pp. 349–358. DOI: 10.1145/2393347.2393399.

45. Yosinski, J., Clune, J., Bengio, Y., Lipson, H. (2014). How transferable are features in deep neural networks? Vol. 2, pp. 3320–3328.

46. Zhou, X., Jin, K., Shang, Y., Guo, G. (2020). Visually interpretable representation learning for depression recognition from facial images. IEEE Transactions on Affective Computing, Vol. 11, No. 3, pp. 542–552. DOI: 10.1109/taffc.2018.2828819.

Article received on 11/01/2023; accepted on 15/03/2024.
** Corresponding author is Luis A. Castro.*

ISSN 2007-9737

Automatic Knowledge-Driven Approach for Optimal Service Selection based on ELECTRE III and Quality-Aware Services

Hela Taktak[1,2,*], Khouloud Boukadi[1], Chirine Ghedira-Guegan[3], Michael Mrissa[4], Faiez Gargouri[1]

[1] University of Sfax, MIRACL Laboratory, Tunisia

[2] University of Lyon, LIRIS UMR5205, France

[3] University of Lyon, Iaelyon School of Management, France

[4] University of Primorska, InnoRenew CoE, Slovenia

{hela.taktak1, chirine.ghedira-guegan}@univ-lyon3.fr, khouloud.boukadi@gmail.com, michael.mrissa@innorenew.eu, faiez.gargouri@usf.tn

Abstract. Optimal real-time collection of a variety of environmental parameters from several environmental data sources, still remains a challenge in the selection process. As environmental web services now have access to a wider range of environmental data sources, the quality of these services can vary, even if they offer the same functionality. This competition among providers means that environmental data may differ in quality. Due to this competition, different environmental data sources compete to provide these functionally equivalent services with different levels of quality: the quality of services (QoS), as well as, the quality of the data sources themselves and their data (QoDS). Therefore, we present an approach to satisfy the need of ranking and selecting the optimal services. Our contribution is an automated knowledge-driven approach that relies on the ELECTRE III MCDM (Multi-Criteria Decision-Making) method and on quality-aware service selection, to optimally select services.

Keywords. Optimal service selection, multi-criteria decision-making (MCDM).

1 Introduction

The environmental data comes from diverse observations sent by traditional sensors (e.g., satellites, sensors, etc.), social media platforms (e.g., cell phones, etc.), or cyber-physical systems. Several challenging issues emerged, since, these data sources have different characteristics, such as the used protocols, access techniques, and data formats.

Actually, due to the diverse characteristics of these data sources, the dynamic change of data on the Web and their related quality metrics over time impacts selecting optimal data sources with their related optimal data, which remains a challenge. The access to these data sources is realized through a layer of data services.

Although the services may offer the same functionality, they can vary in terms of non-functional attributes, such as Quality of

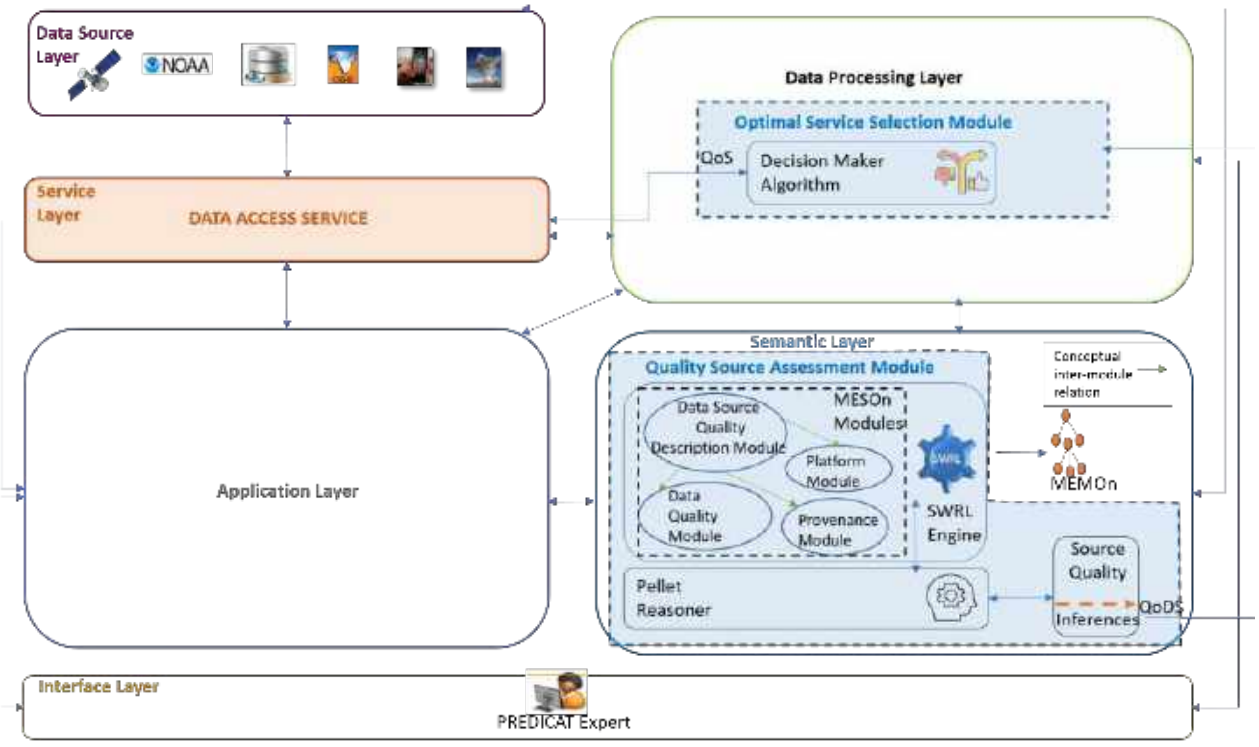


Fig. 1. Overview on the layered architecture of the proposed automatic knowledge-driven solution for optimal service selection

Service (QoS), which includes response time, availability, cost, etc.

For example, when choosing a service, one might prioritize the cheapest option, the fastest option, or perhaps a compromise between the two, that access to several data sources having different quality attributes like trustworthiness, availability, accuracy of the data sources, or also, age and accuracy related to the data itself.

Consequently, many competing services may offer the same concept with different QoS, especially, with a large number of potentially trustworthy services and constantly emerging new services. The primary concern is how to evaluate the quality of environmental data sources and the data they provide.

Otherwise, there is a need to define and explicit the qualities related to the data sources and data. This problem persists as the web environment becomes more dynamic, offering distributed large datasets that require qualification.

The second challenge involves determining the optimal selection of services, which remains a significant issue, specifically, while taking into account the quality related simultaneously to data sources, data and services. Therefore, to tackle these challenges, analyzing competitive qualities and emerging services dynamically requires intelligent analytic techniques.

These techniques provide enhanced decision-making strategies for selecting services. Various approaches have been explored for discovering and selecting web services, primarily relying on ontologies. Examples include OWLS-MX2, WSMO-MX, and SAWSDL-MX2. OWLS-MX2, WSMO-MX, and SAWSDL-MX2 [29, 28, 42].

Although these approaches focus on a better match of the functional or non-functional

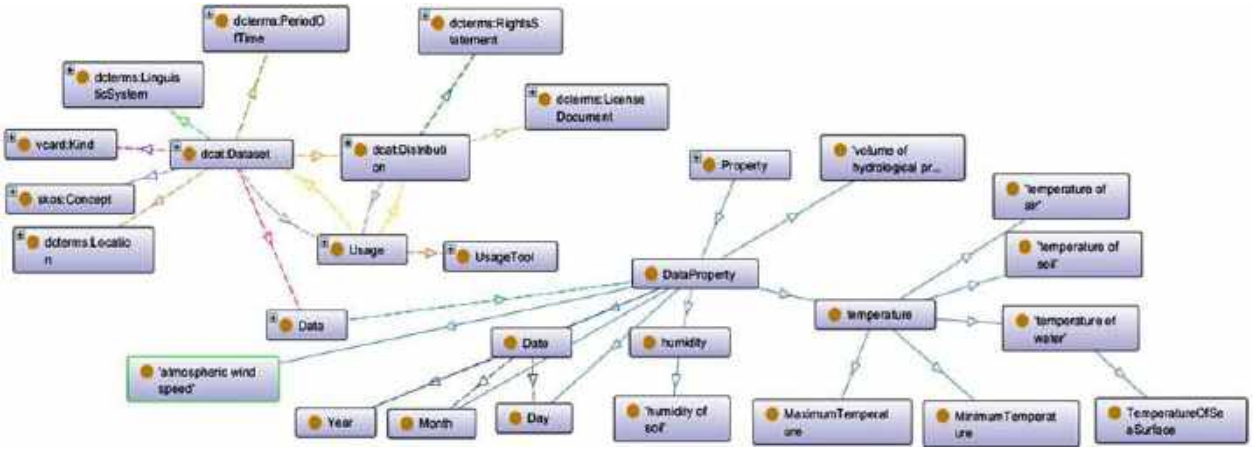


Fig. 2. The data source description module

parameters with the user requirements. They fail to offer a definitive ranking of the optimal selected services, especially when requests involve complex constraints.

These constraints include the quality of the data sources, the data itself, and the quality of services. Several approaches were proposed in the literature, to find the final ranking along with the optimal solutions for the services selection issue related to multi-criteria decision-making (MCDM) techniques. The major advantage of using a multi-criteria method is that it allows modeling the scoring of the optimal solutions, in a more realistic scenario, where a trade-off between conflicting objectives must be resolved.

Several works were proposed to resolve scoring the optimal solutions for the selection problem. Among others, according to [45], ranking approaches such that; AHP, PROMETHEE [14], and ELECTRE [43] are not suitable for directly ranking services due to their high complexity.

For this reason, works such as [37, 36, 53] used the skyline paradigm [12] to search for the optimal dominant services across an important number of services. Skyline solves the selection problem by reducing the search space of services and determines the set of the dominant services based on a Pareto-front.

Nevertheless, it presents two issues: the first one is that its retrieved dominant services are incomparable, without giving any recommendation upon which service to select, thus, causing some confusion in the decision-making process. The second issue, in a large-scale environment, a large number of skyline services could be retrieved with no ranking mechanism.

Therefore, adopting the fuzzy dominance relationship [9] allows us to address both of the stated skyline issues, since it is difficult to classify, re-filter and thus, prune more dominated services in the set of skyline services. To fulfill the aforementioned issues, we propose an automated knowledge-driven solution based on quality-aware selected services. Our value-added contributions can be summarized as follows:

– The first contribution focuses on evaluating the quality of environmental data sources and their inherent data. Our proposal ensures the freshness and reliability of these data sources and their associated data.

  To achieve this, we introduce an ontology that defines quality dimensions and their corresponding inferences for assessing the quality of data sources.

– The second contribution tackles the challenge of optimal service selection with a focus on quality-awareness. We propose an automated knowledge-driven solution for optimal service selection and ranking.

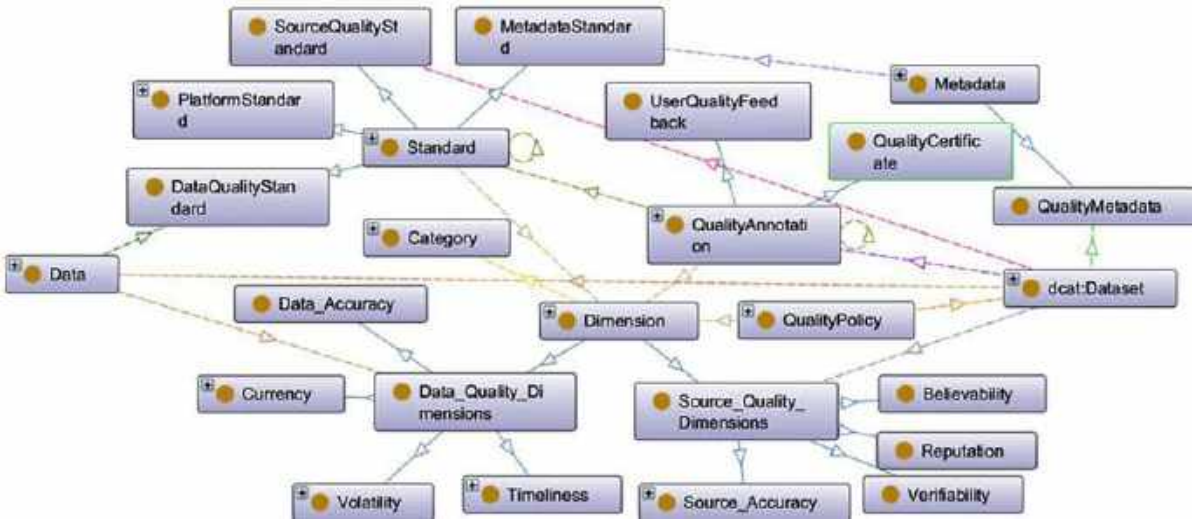


Fig. 3. The data quality module

In addition to considering the Quality of Service (QoS) of candidate services, our approach also takes into account the quality of the data sources (QoDS) as inputs for the skyline operator and the Multi-Criteria Decision-Making (MCDM) technique, specifically ELECTRE III [43].

The skyline operator is employed to reduce the search space of service candidates before proceeding to the ranking process. As for the ELECTRE III MCDM method, it offers optimal rankings of services based on their qualities [17, 16, 38, 22].

The rest of this paper is structured as follows: Section 2 provides a review of related works focusing on QoS-aware solutions using Multi-Criteria Decision-Making (MCDM) methods for optimal service selection and ranking.

Section 3 presents an overview of the layered architecture of our automated knowledge-driven solution for optimal service selection.

Section 4 elaborates on our definition of quality dimensions related to data sources using our proposed modular source ontology, the Meteorological and Environmental Source (MESOn) ontology, along with its related quality dimensions and modules.

Section 5 presents our proposed approach for the optimal service selection, denoted Bα-DSS (Best α-Dominant Skyline Service). Section 6 details the applicability and the evaluation of our approach through several experiments related to the optimal service selection.

Section 7 presents the threats to validity related to our proposals. Finally, we conclude our findings and outline our future work in Section 8.

2 Related Work

This section overviews the most relevant related works about QoS-aware Web services selection, including MCDM approaches, and some other solutions adopted in the service selection task. In the problem of selecting services based on Quality of Service (QoS), quality dimensions, also known as quality criteria, have always been considered crucial because of their direct influence on the selection of optimal services. Various quality dimensions are linked to data sources, including trustworthiness, accuracy, and timeliness. Quality dimensions related to data provided from the data sources are also considered important in the decision-making problems. Moreover, several quality dimensions are linked to the

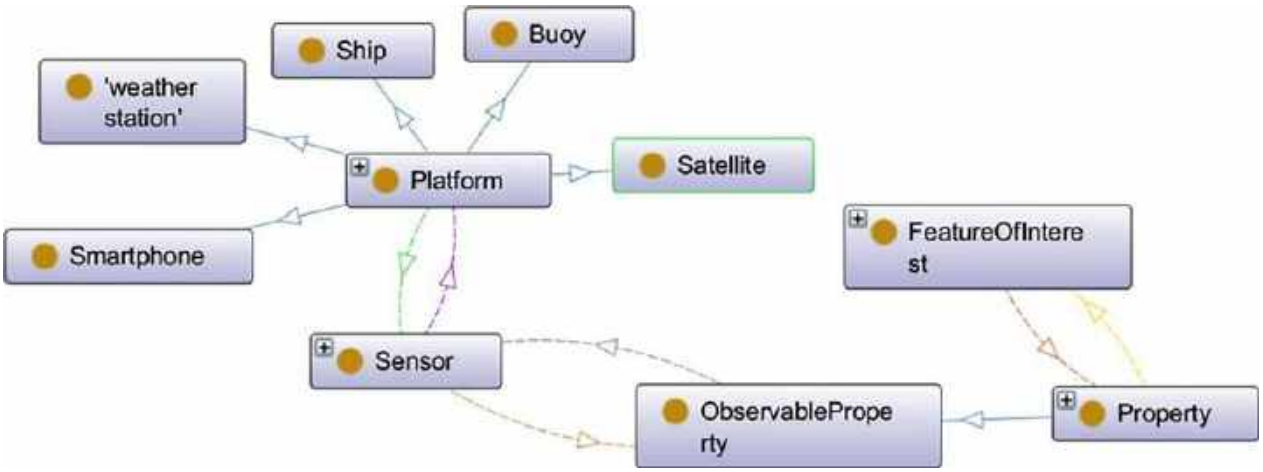


Fig. 4. The platform module

Quality of Service (QoS), including execution time, availability, reliability, reputation, and throughput.

A survey conducted by [8] organizes the building blocks of these quality dimensions into a taxonomy for dataset profiling, including their assessment, summarization, and characterization processes. Dataset profiling, as defined by [8], involves a set of characteristics, both semantic and statistical, that describe a dataset comprehensively, considering the diversity of domains and vocabularies on the Web of data.

However, one of the challenges faced in dataset profiling is computing and interpreting the profiling results. Hence, there is a need for a dedicated solution to reason about and evaluate the dataset used. Ontology stands as a suitable candidate for interpreting and explicitly assessing the quality related to environmental and meteorological data sources. Consequently, to ensure the enrichment of the data sources with quality dimensions, we propose, in this work, the MESOn ontology along with its inferences.

This ontology facilitates dataset profiling and interprets the qualities at both levels: the data source and the data retrieved by the service accessing the data source. Several works dealt with the problem of QoS-aware Web services selection. Authors in [33] and [5] adopted the Linear Programming technique to find the optimal service selection extended with a model evaluating the QoS parameters in [33].

The work in [49] developed a selection algorithm based on QoS evaluation through a QoS evaluation ontology. However, these works consider only a small number of services and QoS parameters, whereas the selection process relies on exponential space complexity. Accordingly, Recent studies concentrate on the skyline algorithm to reduce considerably the important number of services.

Moreover, the application of skyline can be considered as a pre-processing step, since it significantly reduces the search space of the service candidates, and therefore, reduces the computation time when applying the ranking and the selection algorithms.

The skyline concept was firstly introduced in the field of database, by Börzsönyi et al. [12] producing 3408 citations since 2001 (Google Scholar, May 2024). Several algorithms were proposed by Börzsönyi et al. to compute the skyline alternatives built on the Block Nested Loop (BNL) and Divide and Conquer (D&C) [12] algorithms. Other proposed progressive skyline algorithms, which are the Index and Bitmap-based algorithms [47], can output the skyline services without scanning the entire set of the alternatives. Moreover, the Nearest Neighbor (NN) and Branch and Bound Skyline (BBS) algorithms, which rely on the R-tree

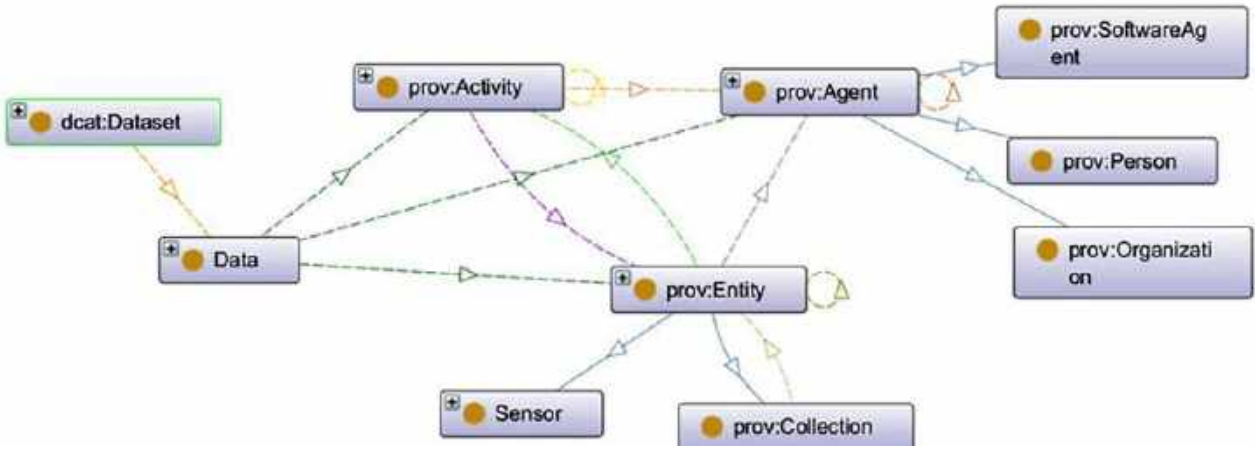


Fig. 5. The provenance module

indexing structure introduced in [39], and that can progressively scan the set of services alternatives.

In order to tackle the problem of large skyline sets, many works such that [54, 9, 17, 16, 38] are proposed, returning K-Representative services to best describe the full skyline set. However, the computation of K-Representative Skyline is a costly problem, since it is based on the multidimensional function. Additionally, the incomparability between service skyline candidates remains an issue in the K-Representative Skyline method.

Therefore, this approach may lack user control over the size of the returned skyline set, especially when dealing with a high number of quality dimensions. Additionally, it does not provide information on the comparative relationship between different skyline service candidates to select the optimal one. Previous studies have relied on the Pareto dominance relationship, as demonstrated in works such as [4, 15, 1]. Furthermore, only a few research works have combined the skyline approach with Multi-Criteria Decision-Making (MCDM) based approaches to solve the QoS-based selection problem and rank the services to select an optimal one. In addition, it is worthy to note, knowledge-driven MCDM methods are only considered, in recent past years. In their work, Dorfeshan et al. [19] introduced a novel data- and knowledge-driven Multi-Criteria Decision-Making (MCDM) method to reduce dependence on expert assessments.

They employed an extended version of the data-driven Decision-Making Trial and Evaluation Laboratory (DEMATEL) method to determine the criteria weights. Additionally, they used the knowledge-driven ELECTRE and VIKOR methods to rank the alternatives.

The Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) method, as described by Zou et al. [56], has been widely utilized as a decision support method in various studies. Its applications include selecting property development locations, cars [57], [6], and mobile applications [26].

In another study, TOPSIS was employed by authors in [52] to optimize service selection on the cloud. Polska et al. [41] developed a web service selection approach based on sensitivity analysis. They compared the Logic Scoring of Preference (LSP) method with other MCDM methods such as Analytic Hierarchy Process (AHP), VIKOR, and TOPSIS.

Sun et al. [46] introduced a fuzzy decision-making framework and MCDM-based approach for cloud service selection. Their work involved the use of a fuzzy ontology to model uncertain relationships between objects in databases for service matching.

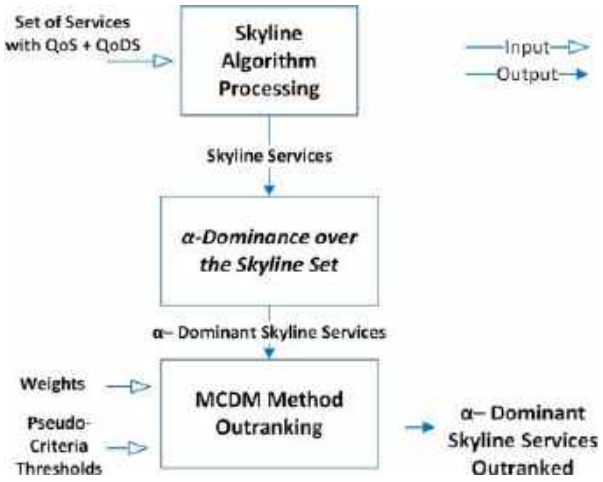


Fig. 6. Bα-DSS steps approach

They employed the AHP method to calculate semantic similarity between concepts and the TOPSIS method for multi-criteria decision-making to rank cloud services. In a related study, Kumar et al. introduced a new framework named CSS-OSSR (Optimal Service Selection and Ranking of Cloud Computing Criteria) [31].

They utilized the TOPSIS method to determine the final ranking of cloud services. Another approach, proposed by Ouadah et al. [37], is called SkyAP-S^3, which integrates skyline with AHP and PROMETHEE to rank services.

Serrai et al. [44] proposed a method that combines skyline with several MCDM (Multi-Criteria Decision-Making) techniques such as SAW, VIKOR, and TOPSIS for service selection and ranking. Xu et al. [51] addressed the QoS-aware service selection problem based on user preferences and fuzzy datasets.

They utilized fuzzy set theory and a fuzzy genetic algorithm to rank web services. In [9], authors proposed the α-dominance principle to rank the Web services based on the quality of services and assigned a fuzzy dominating score to services. However, to our best knowledge, none of these works considered at the same time, the skyline paradigm reducing the services number, the fuzzy degree dominance enabling classifying, re-filtering and pruning more services in the service skyline set, and finally the ranking mechanism.

Although these solutions are a promising direction, efforts are still needed for enforcing and optimizing the quality of solutions. While the existing solutions represent a promising direction, there is still a need for further efforts to enforce and optimize the quality of these solutions.

Our study is related to previous research on the QoS-aware service selection problem, aiming to automatically select optimal services based on their quality dimensions. However, these prior studies overlook the quality of the data sources (QoDS) and fail to account for the dynamically changing service environment.

Furthermore, QoS may be constantly changing, so it is essential that the service selection and ranking be automatic and knowledge-driven. Furthermore, as far as we know, these studies did not emphasize the importance of providing support for describing and inferring the constantly changing quality attributes of environmental data sources.

3 Overview of the Layered Architecture of the Automatic Knowledge-Driven Solution for the Optimal Service Selection

We present an overview of the layered architecture of our proposed automatic knowledge-driven solution for optimal service selection. This solution is developed within the framework of the PREDICAT (PREDIct natural CATastrophes) project, aimed at predicting natural disasters resulting from climate changes[1].

This architecture encompasses seven layers, namely: (1) data source layer, (2) service layer, (3) application layer, (4) user interface layer, (5) semantic layer, and (6) data processing layer. These layers are illustrated in Figure 1.

Our contribution to the automatic knowledge-driven solution for optimal service selection is twofold. Firstly, it includes the Quality Source Assessment Module within the semantic layer. Secondly, it involves the Optimal Service Selection Module situated in the data processing

[1] sites.google.com/view/predicat/predicat.

Table 1. Weight assignment for quality dimensions

Weights	S_ET	S_Av	SO_Acc	QualAssesModule	SO_DTime	QualAssesModule
Assigned Weights$_i$	6	6	4	4	3	1
w_i	0.25	0.25	0.167	0.167	0.125	0.041

layer. These modules are highlighted in blue, in Figure 1.

The data source layer comprises various data sources, including meteorological and environmental observations (EO) from organizations such as NASA, Copernicus, OpenWeather, and others. In the service layer, RESTful (Representational State Transfer - [21]) services are automatically generated to facilitate access to the diverse environmental data sources.

The application layer falls outside the scope of our current work. The user interface layer includes a sophisticated user interface that communicates danger alerts to the experts using the PREDICAT platform.

The semantic layer includes the Meteorological and Environmental Source Ontology (MESOn), which is our first contribution. It also incorporates the Modular Environmental Monitoring Ontology (MEMOn), which comprises a collection of ontological modules addressing various sub-domains within environmental monitoring. MESOn ontology is encompassed in the Quality Source Assessment Module.

This latter evaluates and describes the meteorological and environmental data sources, and their related quality dimensions presented, in section 4. The qualities related to the environmental data sources are captured dynamically.

Moreover, to deduce and analyze the quality dimensions of a specific environmental data source, we employed SWRL (Semantic Web Rule Language) rules along with the Pellet reasoner.

As a result, we obtain the Quality of Data Sources (QoDS) inferences that will be used by the Optimal Service Selection Module and, specifically by our decision-maker algorithm (i.e., Bα-DSS), to select optimally an access service to the assessed data source. These QoDS are queried through the SWRL rules, from the MESOn ontology.

This paper focuses only on the semantics applied for the quality of source assessment through the MESOn ontology. Furthermore, semantics were used in data source profiling according to reasoning on two levels: the quality of the data source itself and the quality of the data returned by the service accessing that data source.

The data processing layer includes our second contribution, which is the Bα-DSS. It deals with the process of selection of the optimal services, through a ranking mechanism. To this end, the data processing layer encompasses the Optimal Service Selection Module.

This module, responsible for identifying optimal service candidates for service composition, employs successive analytical methods to eliminate less important services (i.e., dominated services). Additionally, it utilizes the multi-criteria decision-making method, ELECTRE III, to rank service alternatives.

Furthermore, this module encompasses our proposed decision-maker algorithm referred to as Bα-DSS, that implements the already mentioned analytical methods, taking into consideration the assigned preferences to the criteria (i.e., the criteria related to the quality dimensions QoDS and QoS) specified by experts of the PREDICAT platform.

We provide, in the following, further details on the ontology-based quality assessment to represent the MESOn ontology with its related quality dimensions and modules.

4 Ontology-Based Quality Assessment

The aim of the ontology-based quality assessment process is to assess the quality of both the data sources and the data they provide. The proposed MESOn ontology relies on the Quality Source Assessment Module, encompassed in the semantic layer.

Algorithm 1: Calculate Skyline Services

```
Input: List of Services S, List of Criteria ListCrit
Output: List of Skyline Services Sky
Function ComputeBNLSkyline;
foreach p in S do
    if (Sky = ∅) then
        Sky ← {p};
    foreach q in S do
        Res ← ComparisonFct(p, q, ListCrit);
        if (Res > 0) then
            Sky ← Sky ∪{p};
            S ← S − {q};
        else if (Res < 0) then
            Sky ← Sky −{p};
return Sky;
End Function
```

We detail, in the following, at first, the selected quality dimensions adequate to our requirements describing the data sources and their data quality (QoDS), in addition to the Quality of Service (QoS). Then, we present how to calculate them. Finally, we highlight the semantic aspects through the Quality Assessment Module, which incorporates the MESOn ontology and its related modules.

4.1 Quality Dimensions

Quality dimensions, includes both quality related to data sources and quality related to data. Quality dimensions are commonly conceived as a multidimensional construct, where each dimension represents quality-related characteristic as a multidimensional construct; such as accuracy, timeliness, completeness, relevancy, objectivity, believability, understandability, consistency, and conciseness [3].

Furthermore, Quality dimensions are often grouped into categories known as quality categories. Each quality category comprises one or more computed quality metrics, whose values serve as indicators of quality. According to [27], there are 127 data quality dimensions identified in the literature.

Considering the objectives of our study for the Quality Source Assessment Module, we have selected specific quality dimensions, including source accuracy, and trustworthiness for the data source, and volatility, currency, and timeliness for the data.

4.2 Computing the Quality Dimensions

In order to evaluate the quality dimensions, we describe in the following, our proposal through a formal approach to compute the different retained quality dimensions. Source Accuracy : refers to the percentage of provided values that are consistent with the given gold standard, as described in the literature [7]:

$$\text{SourceAccuracy} = \frac{NG}{NT}, \tag{1}$$

where:
$NG =$ Number of instances of data flagged as good.

$NT =$ Number of total values.

Volatility describes the time period during which information remains valid in the real world, as in [25].

It is the length of time, where data remains valid, as in [7, 40]. Currency, concerns how promptly data are updated with respect to changes occurring in the real world in [7]:

$$\text{Currency} = \text{Age} + \text{DeliveryTime} - \text{InputTime}, \tag{2}$$

where:

- DeliveryTime: Indicates the time when the data are delivered to the user.
- InputTime: Denotes the time when the data are received by the system.
- Age: Represents the age of the data when first received by the system.
- Timeliness: refers to the suitability of the data age for the specific task [48]:

$$\text{Timeliness} = \max\left(0,\ 1 - \frac{\text{Currency}}{\text{Volatility}}\right). \tag{3}$$

- Trustworthiness: The trustworthiness category consists of three dimensions: believability, reputation, and verifiability.

Algorithm 2: Calculation of α-Dominant Skyline Services

```
Input:  α: Degree of dominance
        ϵ: ϵ-value
        λ: λ-value
      Sky: List of Skyline Services
Output: α_Sky: List of α-Dominant Skyline Services
        α_Sky ← ∅
Function
Computeα_DominantSkyServices(α, ϵ, λ, Sky):
    foreach Element in Sky do
        deg ← Compute_Degree_Service(Element,
          NextElement)
        remove(Element)
        if deg ≥ α then
            α_Sky ← List_Of_α_Dominant_Services()
    return α_Sky
End Function.
```

– Believability: Refers to the extent to which data are considered true, real, and credible [7].

– Verifiability: Refers to the degree and ease with which the information can be checked for correctness [11, 7].

– Reputation: is a judgment made by a user to determine the integrity of a source. It can be associated with a data publisher, a person, organization, group of people or community of practice, or it can be a characteristic of a dataset [48, 7].

Due to the correlation of believability, verifiability, and reputation, and for simplification reason, we chose to treat the trustworthiness as a block.

Many authors dealt with trustworthiness by proposing different ways of calculation. We opted to assess trust using two approaches:

Models and tools. For models, we employed the 7Ws Model [23], which involves answering seven questions and then calculating a score between 0 and 7 based on the responses.

More information about these questions is provided in the subsequent sub-section 4.3.2.

4.3 MESOn: A Source Ontology with Quality Dimensions

In order to evaluate the quality related to data sources and their inherent data, to our best knowledge, there is no ontology dedicated to explicit environmental and meteorological data sources and to assess their quality, in order to interpret and exploit this assessment. We tended to use ontology, owing to the fact, to define a shared conceptualization of our problem related to the assessment of the data source qualities.

Hence, we chose to design a data source ontology by reusing some fragments from other ontologies and vocabularies. In this context, we have analyzed the existing ontologies and vocabularies. We examined the available ontologies and vocabularies, and then introduced our MESOn ontology, which includes quality dimensions associated with meteorological data sources.

We incorporated fragments from validated ontologies such as the Dataset Quality Ontology (daQ) [18], Data Quality Vocabulary (DQV) [2], Data Catalogue Vocabulary (DCAT) [34], Data Usage Vocabulary (DUV) [20], PROV-O ontology [32], and SOSA/SSN Ontology.

Consequently, our proposed modular ontology is stable and the reused fragments respect the W3C standards. We present, in the following, the main modules encompassed in our proposed MESOn ontology. Then, we detail how to use inferences to reason on the assessment of the quality of the data sources, in MESOn ontology.

4.3.1 The Source Ontology Modules

We adopted a modular approach, a recognized best practice for developing high-quality ontologies, which facilitates easier maintenance and promotes reusability. Therefore, MESOn is constituted of four modules, detailed in the following.

– The Data Source Description Module Figure 2 details the Data Source Description Module with its related classes.This module focuses on describing the data source, detailing the dataset (dcat:Dataset) and its characteristics.

Table 2. Reasoning time for the MESOn ontology

Data Source	Reasoning Time (ms)
Copernicus	141
NASA	150
OpenWeather	110
NOAA	159
CHIRPS	133
GPCP	139
UCSB Climate Hazard Center	147
OSS	127
HWSD	130

These characteristics include the time period (dcterms:PeriodOfTime), observation locations (dcterms:Location), linguistic system used (dcterms:LinguisticSystem), and the various types of data it contains (vcard:Kind).

This module contains, also, information about the form of the dataset (i.e., the dcat:Distribution). For instance, it describes the type of dataset (e.g., XML dataset, Web service, database, etc.) and its specific data properties such as the URL, username, and password.

Additionally, this module provides information about how the dataset is used, the tools that manipulate it, and the required license for its usage. All these characteristics related to the description of the data source, provide information on what is the format of the dataset and which is the tool to open it.

– The Data Quality Module Figure 3 describes the Data Quality Module with its related classes. This module elaborates on various quality characteristics, encompassing quality dimensions, standards, certificates, quality policies, and user feedback on quality.

  Its primary objective is to evaluate the quality of meteorological and environmental data sources and their associated data, utilizing the SourceQuality and DataQuality classes. We have represented the calculated quality dimensions based on the details provided in sub-section 4.2.

– The Provenance Module Figure 5 depicts the Provenance Module with its related classes. This module reuses fragments from the provenance ontology (i.e., PROV-o). The provenance module provides information about the data lineage, indicating the origins of a data unit. Its main concepts include Entity, Activity, and Agent.

  The Agent class represents the entity responsible for carrying out activities. Agents can be categorized as SoftwareAgent, Person, or Organization. The Activity class illustrates the activities involved in generating the data. These activities are performed by agents and entities.

  The Entity class showcases entities involved with data units. As depicted in figure 5, Sensor is an Entity and Collection is a class, which includes a group of entities (e.g., Sensor Network).

– Platform Module Figure 4 represents the Platform Module The module includes descriptions of platforms capturing meteorological and environmental observations (e.g., temperature) and the sensors they host (e.g., smartphones and satellites).

  Each sensor tracks an observable property and its feature of interest. For example, if we consider air temperature as the observation required, measured by an iPhone, the platform would be a smartphone represented by an individual named “iPhone 9-IMEI 35-207776-824955-0”.

This platform contains a sensor represented by the individual “Bosch SensortecBMA253”. The observable property is “Air Temperature” and its feature of interest is “Earth Temperature”.

In the next sub-section, we detail how to use inferences to reason on the proposed data source MESOn ontology, in order to assess the quality of the EO data sources and their related data (QoDS).

4.3.2 Inferences

Our proposed inferences are related to the source accuracy, currency, volatility, timeliness, and trustworthiness. Source Accuracy: The quality of the source accuracy can be computed along two cases. The first one is when the Quality Control Levels are provided with the data observations.

Therefore, the source accuracy is deduced from the accuracy of all items of the data source. In this case, a coefficient for each Quality Control Level is attributed. If the level 1 of Quality control is checked then, a coefficient of 0.5 is attributed.

For the level 2, the coefficient is 0.75 and for the level 3, the coefficient is 1. We adopted these coefficients according to a gradual logic, which correspond to our requirements. The source accuracy in this case is calculated as following:

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y) ^
numberOfInstances(?x,?nins) ^
numberOfQCLevel1(?x,?nqc1) ^
numberOfQCLevel2(?x,?nqc2) ^
numberOfQCLevel3(?x,?nqc3) ^
swrlm:eval(?res,"((nqc1 * 0.5 + nqc2*
0.75 + nqc3)/nins)",?nqc1, ?nqc2,
?nqc3, ?nins)->accuracy_value(?y,?res)
```

The second case is when no quality control annotations are provided with the observations, in the data source. Therefore, we proposed in the procedures of quality controls to assign one of these flags: (Good, Inconsistent, Doubtful, Erroneous, Missing Data) for each observation. Subsequently, the source accuracy is computed, as defined in sub-section 4.2, in Eq. 1.

```
dcat:Dataset(?x)^
Source_Accuracy(?dim)^
hasSourceQualityDimensions(?x,?dim)^
numberOfFlagCorrect(?x,?ncf) ^
numberOfInstances(?x,?nins) ^
swrlm:eval(?res,"(ncf/nins)",?ncf,
?nins)->accuracy_value(?dim,?res)
```

– Currency: We have adopted the following rule to compute the quality of Currency:

– Currency = Age + DeliveryTime - InputTime [40, 7] which can be translated in our case as following:

– Currency = (CurrentDate - Max (Date_Dataset)) + (CurrentDate - LastModification).

Currency is computed according to two cases in SWRL: The first case is dedicated to assign the value of currency when it is greater than 0.

```
dcat:Dataset(?x) ^ Currency(?y)^
hasDataQualityDimensions(?x,?y)^
terms:PeriodOfTime(?p)^
terms:temporal( ?x,?p)^
temporal:add ( ?currentDate, "now",0,
"Days") ^ end( ?p,?e)^
dataset_modified( ?x,?date_modified)^
temporal:duration( ?duration,
?currentDate,?date_modified,"Days")^
temporal:duration ( ?d, ?currentDate,
?e, "Days")^
swrlb: add (?currency, ?d,?duration)^
swrlb: greaterThan (?currency, 0)->
currency_value (?y, ?currency)
```

The second case is applied, when the obtained currency value is equal to 0 or negative. Subsequently, we assign 0 instead.

```
dcat:Dataset(?x) ^ Currency(?y) ^
hasDataQualityDimensions(?x, ?y) ^
terms:PeriodOfTime(?p) ^ terms:
temporal (?x, ?p) ^
temporal:add(?currentDate,
"now", 0, "Days") ^ end(?p, ?e) ^
dataset_modified(?x,
?date_modified) ^
temporal:duration(?duration,
?currentDate, ?date_modified,
"Days") ^ temporal:duration(?d,
?currentDate, ?e, "Days") ^
swrlb:add(?currency, ?d,
?duration) ^
swrlb:lessThanOrEqual(?currency, 0)->
currency_value(?y, "0.0"^xsd:double)
```

– Volatility: Volatility determines the length of the time data remains valid. We considered the following rule to check the volatility of data:

– Volatility = Currentdate $<$ (LastModification + accuralPeriod).

By applying this rule, we are able to compute the duration between the current date and the date of the last modification plus the accuracy, to get the remaining validity period. This computation is achieved following two rules: The first case when the obtained volatility value is positive.

```
dcat:Dataset(?x) ^ Volatility(?y)^
hasDataQualityDimensions(?x, ?y)^
temporal:add(?currentdate, "now", 0,
"Days")^dataset_modified(?x,?datmod)^
dataset_accuralPeriodicity(?x,?ap)^
temporal:add(?datadd, ?datmod, ?ap,
"Days")^temporal:duration(?duration,
?datadd,?currentdate,"Days")^
temporal:before(?currentdate,?datadd)
->volatility_value(?y,?duration)
```

The second case is when the volatility is less or equal to 0. We assign 1 as a value in order to avoid division by 0 in the timeliness rule.

```
dcat:Dataset(?x) ^ Volatility(?y) ^
hasDataQualityDimensions(?x, ?y) ^
temporal:add(?currentdate,"now", 0,
"Days")^dataset_modified(?x,?datmod)^
dataset_accuralPeriodicity(?x, ?ap)^
temporal:add(?datadd,?datmod,?ap,
"Days" )^temporal:duration(?duration,
?datadd, ?currentdate, "Days") ^
temporal:notBefore(?currentdate,
?datadd) -> volatility_value(?y,
"1.0" ^^xsd:double)
```

– Timeliness: Timeliness determines how current the data are for the task at hand [7]. We considered the following rule for the data:

– Timeliness = Max(0, 1 - (Currency / Volatility) [7, 40].

– To compute the Timeliness, we considered two SWRL rules depending on the cases:

– The first case is when the result of timeliness is less than 0. Therefore, the value must be equal to 0. This issue was resolved according to the following rule:

```
dcat:Dataset(?x) ^ Timeliness(?y) ^
Currency(?xc) ^ Volatility(?xv) ^
hasDataQualityDimensions(?x, ?y) ^
hasDataQualityDimensions(?x, ?xc) ^
hasDataQualityDimensions(?x, ?xv) ^
currency_value(?xc, ?c) ^
volatility_value(?xv, ?v) ^
swrlm:eval(?z, "(c / v)", ?c, ?v) ^
swrlb:subtract(?t, 1, ?z) ^
swrlb:lessThan(?t, 0) ->
timeliness_value(?y, 0)
```

The second case is when the value of timeliness is greater than 0. The obtained value is taken. The rule is as follows:

```
Dataset(?x) ^ Timeliness(?y) ^
hasDataQualityDimensions(?x,?y) ^
currency_value(?x, ?c) ^
volatility_value(?x, ?v) ^
swrlm:eval(?z, "(c / v)", ?c, ?v) ^
swrlb:subtract(?t, 1, ?z) ^
swrlb:greaterThanOrEqual(?t, 0) ->
timeliness_value(?y, ?t)
```

Trustworthiness: To reason on the trustworthiness, we used the 7Ws Model [23], consisting on replying to 7 questions. The rationale behind using this model is that the provenance information related to the assessment of the trustworthiness can be identified by answering the seven questions, detailed in the following. We, therefore, created the inferences rules related to the questions of the 7Ws Model.

To compute a score ranging from 0 to 7, we base it on the answers provided. The questions are the following: We check for each question, if it is answered, by assigning a boolean value to each question: 1 as a score if the question is answered (true) or 0 in the opposite case (false).

– What is the name of the author or organization that created the dataset?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
author_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_author(?z,"1.0"^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
author_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_author(?z,"0.0"^^xsd:double)
```

– What is the data?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
data_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_whatis(?z,"1.0"^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
data_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_whatis(?z,"0.0"^^xsd:double)
```

– Which instruments were used to collect the dataset?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)
dqv:inCategory(?y, ?z) ^
instruments_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_instruments(?z,"1.0"
^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
instruments_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_instruments(?z,"0.0"
^^xsd:double)
```

– What events led to the collection of the dataset and how was it collected?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
collected_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_how (?z,"1.0"^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
collected_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_how (?z,"0.0"^^xsd:double)
```

– Why the dataset is created?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
reason_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_why (?z, "1.0"^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
reason_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_why (?z, "0.0"^^xsd:double)
```

– When was it collected?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
when_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_when (?z, "1.0"^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y) ^
dqv:inCategory(?y, ?z) ^
when_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_when (?z, "0.0"^^xsd:double)
```

– Where was it collected?

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
where_b(?x, ?b) ^
swrlb:equal(?b, true) ->
score_where (?z,"1.0"^^xsd:double)
```

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
```

```
dqv:inCategory(?y, ?z) ^
where_b(?x, ?b) ^
swrlb:equal(?b, false) ->
score_where (?z,"0.0"^^xsd:double)
```

After replying to the questions, we calculate the overall score which is the ratio between the answered questions and the total questions number by applying the following rule:

```
dcat:Dataset(?x) ^
hasSourceQualityDimensions(?x,?y)^
dqv:inCategory(?y, ?z) ^
score_why(?z,?s3)^score_where(?z,?s6)^
score_whatis (?z, ?s2) ^
score_when(?z,?s5)^score_how(?z,?s4)^
score_author(?z,?s1)^
score_instruments(?z, ?s7) ^
swrlm:eval (?res,"((s1+s2+s3+s4+s5+s6
+s7)/7)", ?s1, ?s2, ?s3, ?s4, ?s5,
?s6, ?s7) ->
trustworthiness_value (?z, ?res)
```

5 Optimal Service Selection

The automatic knowledge-driven solution for optimal service selection aims at selecting the most appropriate services based on the quality of the data sources, the data itself, and the services.

Our automatic knowledge-driven solution for optimal service selection relies on the data processing and the semantic layers, accordingly, as depicted in Figure 1. To achieve this, we focused our proposed approach, on the one hand, on analytical filtering techniques to reduce the search space of services (i.e., skyline and α-dominance), and on the other hand, on an outranking method, in order to select the optimal ranked service, for a given concept.

The analytical filtering techniques are encompassed in the Optimal Service Selection Module, in the data processing layer. In this section, we present, at first, the skyline approach, a formalization of our concepts using the α-dominance principle, which is based on a dominance relationship combined with the fuzzy sets theory. Second, considering the aforementioned solutions, we hence, base our optimal service selection solution on the Best_α-Dominant_Skyline_Service (Bα-DSS) approach, presented hereafter.

5.1 Background on the adopted Skyline and Fuzzy Sets

As aforementioned in the introduction section, two mechanisms were used: The skyline and the fuzzy sets theory.

A) Skyline: The skyline operator allows retrieving all non-dominated and best alternatives based on a crisp multi-criteria comparison. According to the Pareto sense. One service dominates another, if and only if, it is at least as good as the other in all criteria and better in at least one criterion.

– **Definition 1. (Pareto Dominance)**

Let S = (S_1, S_2, ..., S_n) be a set of n-dimensional services which are functionally similar. The N dimensions are the number of the considered quality criteria.

Let S_i and S_j two services of S. S_i dominates S_j, in Pareto sense, if and only if, S_i is better or equal to S_j in all dimensions and (strictly) better than S_j in at least one dimension.

We assume that a greater value is preferable in each criterion to maximize and a smaller value is preferable in each criterion to minimize. Each service S_i is characterized by a vector $Q(S_i) = (q_1(S_i), ..., q_d(S_i))$ where $q_\iota(S_i)$ denotes the value of the ι-th quality criteria related to the service S_i:

$$\forall \iota \in [1, d], q_\iota(S_i) \geq q_\iota(S_j) \wedge \exists \kappa \in [1, d], \\ q_\kappa(S_i) \succ q_\kappa(S_j). \tag{4}$$

Since the comparison of the quality criteria related to data sources and services are susceptible to the uncertainty, we introduce in the following the fuzzy sets theory.

Algorithm 3: Pseudo-Algorithm for Best α-Dominant Skyline Service (Bα-DSS)

```
Input: List of Services S, α (default value is 0.7)
Output: Best-ranked service Best_Ranked_Service
Function Best_α_DominantSkyService
SKY ← ∅
foreach element in S do
    Sky ← ComputeBNLSkyline();
    α_Sky ← Computeα-DominantSkyServices();
    foreach element in α_Sky do
        Best_Ranked_Service ← ELECTRE III()
return Best_Ranked_Service // The 1st-ranked service
 from the set of ranked services.;
End Function.
```

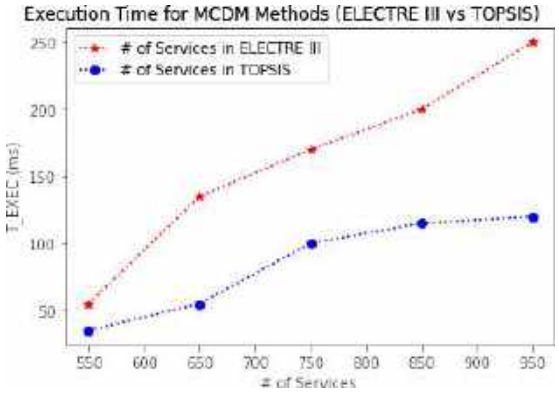


Fig. 7. Execution time for ELECTRE III and TOPSIS methods before applying skyline and α-dominance

A) Fuzzy Sets: We introduce in this part, the fuzzy sets theory and the fuzzy dominance (i.e., α-dominance). Fuzzy sets theory was first introduced in 1965, by Zadeh [55].

The usefulness of the fuzzy sets theory consists of representing vague and uncertain data. The fuzzy logic models uncertain systems to reason and help the decision-making process, when precise information is lacking.

Zadeh defines a fuzzy set as a group of objects with a range of membership grades. The rationale is that an object can belong to a set partially. Moreover, the set can be defined by a generalized membership function that assigns a degree of membership to each object, typically ranging from zero to one.

In the context of skyline computation, fuzzy sets were used to express fuzzy dominance degrees. However, there is no information available on the comparison relationship between candidates in the skyline set of services. Thus, as a first effort, the proposed querying syntax is extended with user-defined fuzzy comparators is SQLf in [13].

Another study in [30], demonstrated the effectiveness of fuzzified Pareto dominance and its application in Evolutionary Multi-Objective Optimization. In order to determine a graded dominance relationship between the different services, we define below the fuzzy dominance relationship based on a specific comparison function that expresses a graded inequality of the type “strongly greater than”.

– **Definition 2. (Fuzzy Dominance)**

Given two services S_i, $S_j \in$ S, we define the fuzzy dominance, as stated in [9], to express the degree to which a service S_i dominates a service S_j as:

$$\deg_{\mu_{\epsilon,\lambda}}(S_i \succ S_j) = \frac{\sum_{\iota=1}^{d} \mu_{\epsilon,\lambda} q_\iota((S_i), q_\iota(S_j))}{d}, \quad (5)$$

where $\mu_{\epsilon,\lambda}$ is a membership monotone comparison function that expresses the extent to which $q_\iota(S_i)$ is significantly greater or lesser than $q_\iota(S_j)$. The membership function $\mu_{\epsilon,\lambda}$ can be defined absolutely way (i.e., in terms of $x - y$) as follows:

$$\mu_{\epsilon,\lambda}(x,y) = \begin{cases} 0 & \text{if } x - y \leq \epsilon, \\ 1 & \text{if } x - y \geq \lambda + \epsilon, \\ \dfrac{x - y - \epsilon}{\lambda} & \text{otherwise}, \end{cases} \quad (6)$$

where $\lambda > 0$, i.e., $\succ$ gives more grade information than the idea of “strictly greater” and ϵ must be ≥ 0. The semantics of $\mu_\succ$ are given in

the following way: x is not significantly greater than y, when $(x - y) < \epsilon$. And x is significantly greater than y, when $(x - y) > \lambda + \epsilon$.

And x is greater than y to some extent, when $\epsilon < (x - y) < \lambda + \epsilon$. λ and ϵ are subjective parameters that are user-defined and domain-specific. They represent the semantics of the gradual relation μ within a particular domain for a specific user, as defined in [10].

– **Definition 3. (α-Dominance):**

In this section, we introduce the concept of the α-dominant service skyline, which is based on the notion of α-dominance, representing a graded form of dominance. For two services S_i, $S_j \in$ S and $\alpha \in$ [0, 1], we state that S_i α-dominates S_j (or S_i dominates S_j to a degree α) in the context of μ_ϵ, λ, denoted as $S_i \succ {\mu_\epsilon,\lambda}^\alpha S_j$, if and only if deg $\mu_{\epsilon,\lambda}$ ($S_i \succ S_j$) $\geq \alpha$. Otherwise, the α-dominance eliminates all the services with a degree below to the fixed α degree value.

5.2 Best_α-Dominant_ Skyline_Service (Bα-DSS) Approach

Figure 6 illustrates the three primary steps of our proposed approach for effectively selecting the optimal service from the initial set of services.

Our proposed Bα-DSS approach is enacted by the decision-maker algorithm encompassed in the Optimal Service Selection Module, within the data processing layer of the PREDICAT platform. The main objectives of our proposed approach, ensured by the decision-maker are as follows:

1. Reduce the overall number of services to decrease the search space and computational time using the skyline algorithm.
2. Support the comparison between alternatives in the retrieved dominant skyline set by applying a fuzzy degree of dominance to eliminate skyline services that do not meet the specified degree of dominance.
3. Apply an outranking mechanism to the compared set of α-dominant skyline services based on a multi-criteria decision-making method (MCDM).

To do so, Bα-DSS is composed of three main steps: the first one is performing the skyline algorithm, which is processed upon a group of functionally similar services, each characterized by various Quality of Service (QoS) and Quality of Data Source (QoDS) criteria, to retrieve the dominant set of skyline services.

This set serves as the input for the next step, which involves applying an additional filter based on the definition of a fuzzy degree of dominance. It consists on performing the fuzzy dominance principle through the α-dominance degree, computing the set of the α-dominant skyline services.

Consequently, the skyline services with a degree of dominance lower than the specified threshold are discarded. Otherwise, the combined use of skyline and α-dominance effectively eliminates dominated services. The skyline and the α-dominance help eliminate dominated services. This reduces the search space for ranking services. It is particularly useful when dealing with a large number of services.

The α-dominant skyline services are the output of the second step of our approach, that will be the input for the next step. Finally, the third step involves an outranking mechanism applied to the set of compared α-dominant skyline services, based on multi-criteria decision-making (MCDM) method. The output is an ordered set of the α-dominant skyline services.

The first outranked service will be selected as the optimal service responding to a fixed requested functional concept. We give, in the following, more details about the three steps related to our proposed Bα-DSS approach.

A) Skyline-based Services Filtering We considered six quantitative quality dimensions: Execution Time $(\mathrm{S\_ET})$, Availability $(\mathrm{S\_Av})$, and Cost $(\mathrm{S\_Cost})$, which are related to the quality of service (QoS), and Accuracy $(\mathrm{SO\_Acc})$, Trustworthiness $(\mathrm{SO\_Trust})$, and Data Timeliness $(\mathrm{SO\_DTime})$, which are related to the quality of data sources (QoDS).

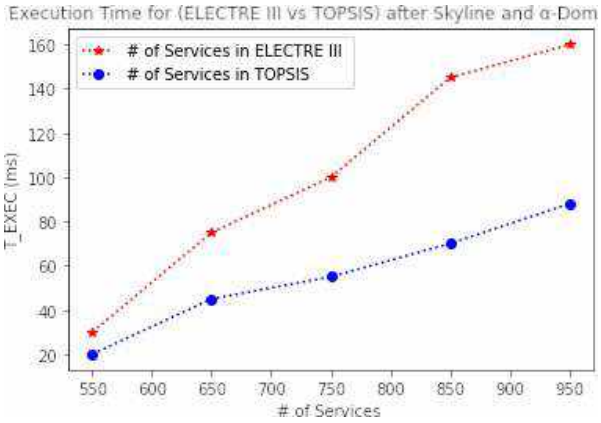


Fig. 8. Execution time for ELECTRE III and TOPSIS methods after applying skyline and α-dominance

We employed the BNL (Block-Nested- Loops) skyline algorithm to compute the set of dominant services due to its popularity and ease of use. The function ComparisonFct(p, q, ListCrit) in Algorithm 1 compares the two services p and q pairwise, considering all the criteria listed in ListCrit.

This function returns a count of the maximum number of criteria for a given service. Furthermore, the main function ComputeBNLSkyline, in Algorithm 1, retrieves the set of all the dominant services in the Sky list.

B) α-Dominant-based Services Filtering as aforementioned, we operate in a fuzzy environment, since the incomparability between the skyline service candidates remains an issue.

As a result, this second step identifies all skyline services that meet the condition of having a fuzzy dominance degree greater than or equal to the specified threshold. To compute the α-Dominant Skyline Services, our proposed algorithm 2 uses the previously detailed functions (e.g., Eq. 5 and Eq. 6). Changes in the α parameter affect the size of the resulting α-dominant skyline services.

Increasing (or decreasing) α includes (or excludes) services with lower-quality compromises. In our study, we varied the α parameter while fixing its value at 0.7. Even if the set of α-dominant skyline services may contain services with a bad compromise, they will be classified and outranked in the MCDM ELECTRE III method.

Moreover, adjusting the values of λ and ϵ enables the retention of services with a satisfactory compromise between the QoS attributes. As λ and ϵ are subjective parameters, we experimented with different values and ultimately set them to 0.2 and 0.1, respectively. These values consistently returned α-dominant services with a desirable compromise between the QoS attributes.

Furthermore, we advocate to use the α degree to 0.7, ϵ to 0.1, and λ to 0.2. These parameters yielded favorable results concerning the α-Dominant Skyline Services set, which demonstrates a satisfactory compromise between QoS attributes.

Additionally, the algorithm computes the dominance degree for each service and verifies whether the dominance degree between every pair of services is greater than or equal to the predefined α-dominance threshold. Subsequently, the algorithm retains all services with a dominance degree greater than or equal to 0.7, constituting the α-dominant skyline services set.

The next section addresses the remaining issue of ranking the α-dominant skyline services. It introduces a ranking mechanism using the ELECTRE III method, to produce an ordered services set, helping in the selection process of the optimal services. We detail in the following, the ranking mechanism.

C) ELECTRE III-based Services Outranking Different versions of ELECTRE were developed (e.g., ELECTRE I, II, III, IV, and TRI). To address the ranking problem of candidate services effectively, we opted for the ELECTRE III method (Roy, 1990) [43]. ELECTRE III was chosen for its capability to handle inaccurate, imprecise, and uncertain comparisons.

The method is based on pseudo-criteria, which act as thresholds, accommodating the uncertainty and ambiguity inherent in calculations and performance evaluations. These thresholds enable fuzzy comparisons, allowing the method to

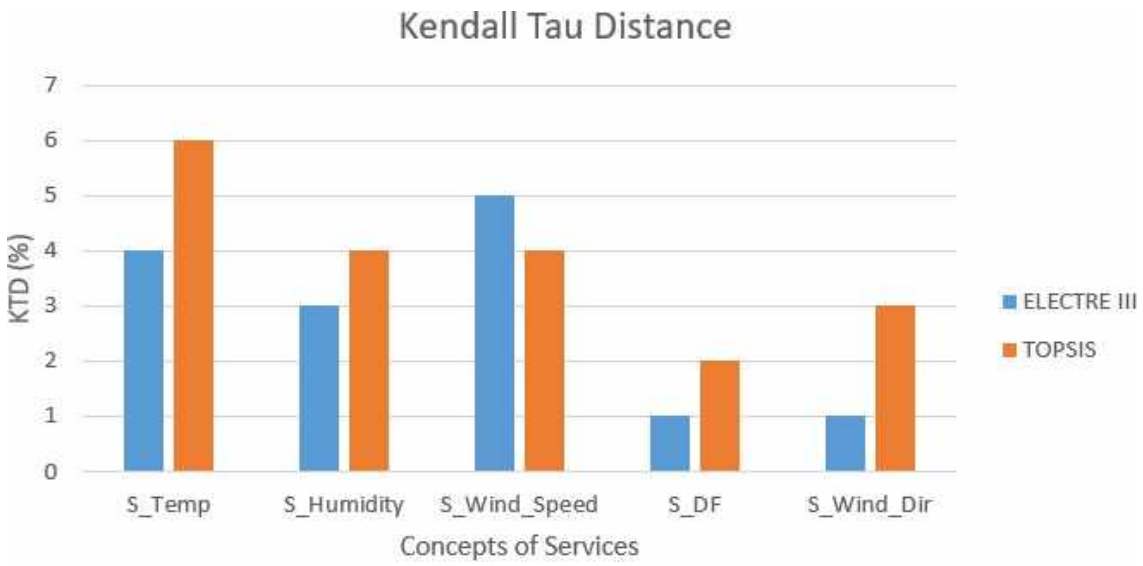


Fig. 9. Kendall Tau Distance (KTD) over ELECTRE III and TOPSIS rankings

draw conclusions based on the set of α-dominant skyline services, which are then ranked.

By exploiting knowledge derived from the quality dimensions (i.e., the criteria in ELECTRE III) of the QoDS inferred from the MESOn ontology, fuzzy comparisons are made, producing valuable decisions. In addition to outline the decision maker's preferences, ELECTRE III assigns weights and pseudo thresholds to each quality criterion.

It serves as a decision-maker to select the best compromise among all considered service alternatives and their criteria. The method is based on pairwise comparisons of alternatives, considering the extent to which evaluations of the alternatives and preference weights confirm or contradict the dominance relationship between pairwise alternatives.

In our case, the quality criterion j can be one of six quality dimensions (i.e., QoS: $\mathrm{S\_ET}$ (service execution time), $\mathrm{S\_Av}$ (service availability), $\mathrm{S\_Cost}$ (service cost), QoDS: $\mathrm{SO\_Acc}$ (source accuracy), $\mathrm{SO\_Trust}$ (source trustworthiness), $\mathrm{SO\_DTime}$ (source data Wtimeliness).

For each criterion, we defined three different pseudo-criteria: The preference threshold (p), the indifference threshold (q), and the veto threshold (v). Experts must specify values related to these thresholds for each criterion, ensuring that $(v \geq p \geq q)$, and assign an importance weight (w_j) for each criterion j, as depicted in Table 1.

PREDICAT experts assigned the important weights for $\mathrm{S\_ET}$, $\mathrm{S\_Av}$, $\mathrm{S\_Cost}$, $\mathrm{SO\_Acc}$, $\mathrm{SO\_Trust}$, and $\mathrm{SO\_DTime}$. We then normalized the criteria weights using the Weighted Arithmetic Mean, as shown in Eq. 7. This normalization ensures that the sum of the weights is equal to 1.

The Weighted Arithmetic Mean is calculated using the following formula:

$$x'_w = \frac{\sum_{i=1}^{n}(w_i x_i)}{\sum_{i=1}^{n}(w_i)}, \tag{7}$$

where :

x_w = is the weighted mean,
w_i = is the allocated weighted value, and
x_i = is the observed value of each criterion.

ELECTRE III method encompasses several steps such that:

1. Estimation of concordance indices,

Table 3. Parameter configurations for the penalty-based GA

Attribute	Value (Condition)
The Size of Population	100
Initial Population	Solutions randomly generated
Probability of Crossover	0.8
Probability of Mutation	0.1
Termination condition	No enhancement observed in the optimal individual for 30 consecutive generations

2. Estimation of discordance indices,
3. Estimation of credibility scores,
4. Performing distillation procedures, and
5. Performing the complete ranking.

The relation a outranks b, denoted aSb, is asserted by measuring the concordance and the discordance indices. In our case, a and b are the pairwise alternatives of services to be compared. We unrolled the first step of computing the concordance index by comparing the performance alternatives over each criterion individually.

This comparison is weighted, and the formula for $c_j(a,b)$ is given by Eq. 10. For example; $c_j(a,b)$ is the concordance index computed for both services a and b, which are S_1 and S_2 respectively, and which are responding to the same functional concept (e.g., temperature):

$$C(a,b) = \frac{1}{W}\sum_{j=1}^{d} w_j c_j(a,b), \tag{8}$$

where j: criterion, d: the number of the used criteria, w: the used weight corresponding to its criterion from table 1, (a and b) are the services, where:

$$W = \sum_{j=1}^{d} w_j. \tag{9}$$

And following these cases:

$$c_j(a,b) = \begin{cases} 1 \quad \text{if} \quad g_j(a) + q_j(g_j(a)) \geq g_j(b), \\ 0 \quad \text{if} \quad g_j(a) + p_j(g_j(a)) \leq g_j(b), \\ \dfrac{g_j(a) - g_j(b) + p_j(g_j(a))}{p_j(g_j(a)) - q_j(g_j(a))} \text{ otherwise,} \end{cases} \tag{10}$$

where: $g_j(a)$ and $g_j(b)$ correspond respectively, to the performance retrieved values of the quality dimension j of the services a (i.e., S_1) and b (i.e., S_2), respectively, which are responding to the same functional concept (e.g., temperature):

$p_j(g_j(a))$ corresponds to the assigned preference threshold to the performance value of the quality dimension j for the service alternative a and $q_j(g_j(a)$ corresponds to the assigned indifference threshold to the performance value of the quality dimension j for the service alternative a. The first case, when $c_j(a,b) = 1$, means that alternative a is at least as good as alternative b, with the possibility of being better, by a margin equal to the indifference threshold for criterion j.

In the second case, if $c_j(a,b) = 0$, the alternative a is considered not better than alternative b for criterion j. Otherwise, the relationship is between these two extremes.

Then, we unrolled the second step, which consists of computing the discordance index for each pair (i.e., pairwise) of alternatives a and b, for each criterion j, according to Eq. 11. The discordance index expresses the extent to which the concordance index is weakened in the

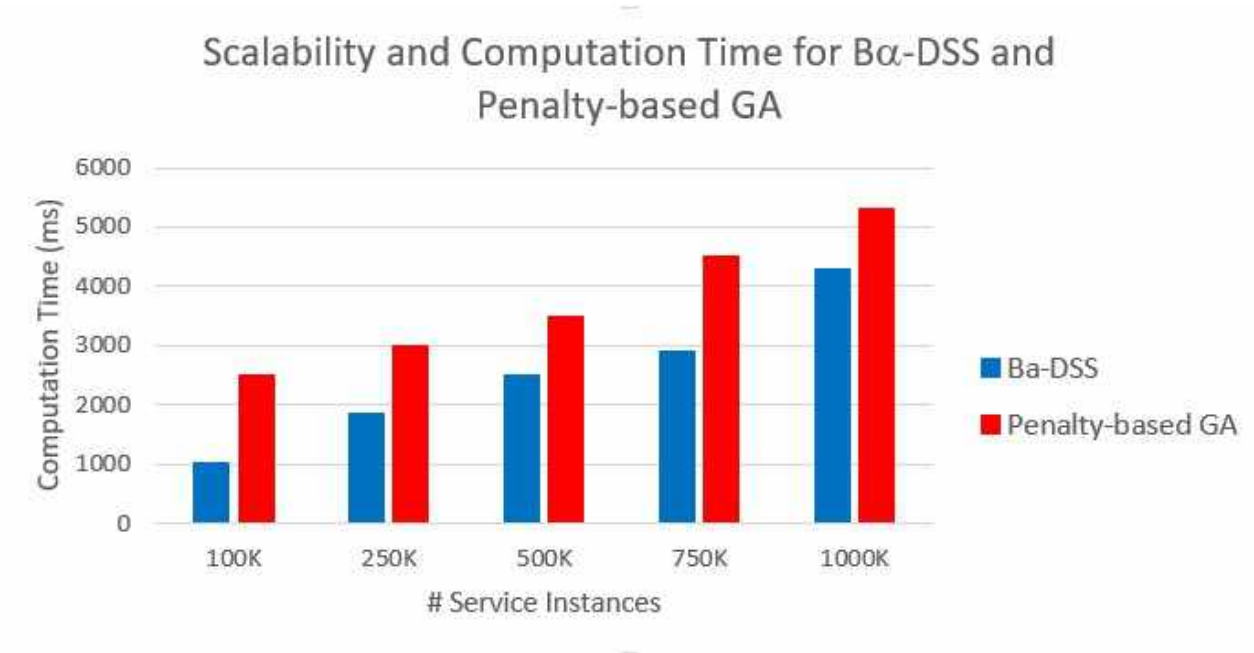


Fig. 10. Scalability and computation time (ms) for Bα-DSS and the penalty-based GA

outranking relations. It verifies the case where an alternative a (i.e., S_1) is worse than b (i.e., S_2).

It is based on the veto (v) threshold. The veto threshold for criterion j is the value from which to refuse any credibility favoring the outranking of the alternative a by alternative b, even if all the other criteria are in concordance with this outranking:

$$D_j(a,b) = \begin{cases} 1 \quad \text{if} \quad g_j(b) \geq g_j(a) + v_j(g_j(a)), \\ 0 \quad \text{if} \quad g_j(b) \leq g_j(a) + p_j(g_j(a)), \\ \dfrac{g_j(b) - g_j(a) - p_j(g_j(a))}{v_j(g_j(a)) - p_j(g_j(a))} \quad \text{otherwise}, \end{cases} \tag{11}$$

where: j is a criterion, $g_j(a)$ and $g_j(b)$ correspond to the performance retrieved values of the quality dimension j of the services a and b, respectively. $p_j(g_j(a)$ corresponds to the assigned preference threshold to the performance value of the quality dimension j for the service alternative a. $v_j(g_j(a)$ corresponds to the assigned veto threshold to the performance value of the quality dimension j for the service alternative a.

Additionally, in the third step, we calculate the credibility score based on the concordance and discordance indices. This score indicates the degree of credibility of the outranking, depending on two scenarios. The first case occurs when no veto threshold is applied, as described in Eq. 12:

$$S(a,b) = C(a,b) \text{ if } D_j(a,b) \leq C(a,b), \forall j, \tag{12}$$

where: $S(a,b)$ is the outranking relation between the services alternatives a and b (i.e., S_1 and S_2 respectively). The second case when the level of discordance increases above a threshold value, the degree of outranking is determined by the concordance index with a reduction according to the discordance index when no veto threshold is applied, following the Eq. 13:

$$S(a,b) = C(a,b) \prod_{j \in \psi(a,b)} \frac{1 - D_j(a,b)}{1 - C(a,b)}, \tag{13}$$

where $\psi(a,b)$ represents the set of criteria for which the discordance index $D_j(a,b)$ is less than the concordance index $c_j(a,b)$. Then, as a fourth step, we performed the distillation procedures. To do so,

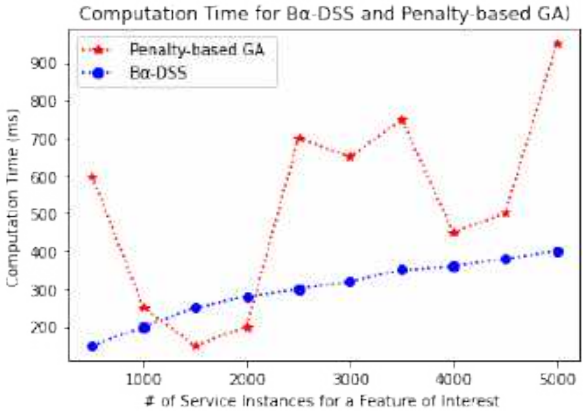


Fig. 11. Computation time (ms) for the Bα-DSS and the penalty-based GA

two iterative processes are generated to obtain two different complete pre-orders.

The first pre-order is descendant (Descendant Distillation), which selects the best alternatives initially and proceeds to the worst. The second pre-order is ascendant (Ascendant Distillation), which selects the worst alternatives initially and proceeds to the best.

Finally, depending on the resulting distillation procedures, we generated a complete ranking of the services alternatives, which are in our case, the α-dominant skyline services. The complete ranking is retrieved by the combination of the previously resulted distillation procedures.

Algorithm 3 provides details on the optimal service selected from the set of α-dominant skyline services after applying the ELECTRE III MCDM method for ranking. This algorithm returns the best-ranked service.

6 Implementation and Evaluation

Below, we outline the implementation and evaluation details of our proposed knowledge-driven solution, focusing on quality-aware service selection for optimal service ranking. We relied on the Protégé-OWL development environment for the reasoning on the quality of the data sources through the SWRL rules. Next, to rank and select the optimal QoS-aware services, we implemented the Bα-DSS method using Java.

The dataset used in Bα-DSS initially consisted of 6 sets of services (concepts). Each set consisted of 500 functionally equivalent services with different QoS attributes corresponding to a given concept. To select the optimal services for a given concept, we used the ELECTRE III MCDM method implemented in Java.

Assigning weights to each quality dimension is a prerequisite for the ELECTRE III method. The assigned weights for each quality dimension (QoS and QoDS) are shown in Table 1. These weights were normalized using the Weighted Arithmetic Mean formula Eq. 7.

6.1 Metrics for Evaluation

In this section, we present three experiments conducted to evaluate and analyze: (1) the reasoning time for data source quality in the MESOn ontology, (2) the effectiveness of our proposed Bα-DSS approach, comparing the relevance of ELECTRE III MCDM ranking results with those of the TOPSIS MCDM method, (3) the complexity assessement of our proposed Bα-DSS approach compared with Penalty-based GA, in terms of the execution time, and (4) the scalability of the Bα-DSS approach compared with Penalty-based GA by varying the dataset size of the candidate services.

6.2 Experiment 1: Reasoning Time for Data Source Quality

We assessed the quality of data sources using SWRL rules for semantic reasoning, as discussed in sub-section 4.3.2. By executing various SWRL rule queries across different environmental data sources, we evaluated the time required for data source quality reasoning. These queries were conducted using the Pellet reasoner within the Protégé 5.5.0 ontology editor. The results of this evaluation are presented in Table 2.

For instance, the reasoning time for Copernicus and NASA data sources was found to be 141 ms and 150 ms, respectively, which are reasonable durations. The execution time of SWRL queries

for all data sources remained consistently low and reasonable. Therefore, the quality of data sources cannot change within this timeframe. As a result, semantic reasoning is unlikely to lead to the misselection of inappropriate data sources by the data processing layer in real-time scenarios.

6.3 Experiment 2: Execution Time and Ranking Performance of Bα-DSS

This experiment serves two purposes: (i) Evaluating the execution time of the ELECTRE III MCDM method compared with the TOPSIS MCDM method [35], with and without applying the skyline operator and the α-dominance. (ii) Assessing the ranking performance of the ELECTRE III compared with the TOPSIS, using the Kendall Tau Distance (KTD) [50].

We employed TOPSIS due to its ability to identify the best α-dominant skyline service alternatives by minimizing the distance to the positive ideal solution (i.e., service) and maximizing the distance to the negative-ideal solution. TOPSIS was used for benchmarking and ranking purposes according to [22].

The initial set of services was expanded to 950 services. The search space was reduced to 500 services by applying the skyline operator and the α-dominance.

As a result of using the skyline and the α-dominance methods, we observed a reduction in the execution time of both the ELECTRE III and TOPSIS, as depicted in figures 7 and 8.

These results demonstrate that employing the skyline and the α-dominance methods is crucial for pruning the dominated services before performing the ranking step through the MCDM method.

Reducing the search space of the services allows us to operate only on the most relevant services, simplifying the selection process.

To assess the rankings produced by the ELECTRE III and TOPSIS MCDM methods, we enlisted the help of environmental experts from the Observatory of Sahara and Sahel (OSS), our socio-economic partner in the PREDICAT project. We proposed 500 ratings of the service candidates (i.e., the 1^{st}-ranked services) to these experts. They were divided into four groups, with each group evaluating approximately 125 ranked service alternatives.

A cross-validation process was then conducted among the different groups. We measured the Kendall Tau Distance (KTD) coefficients between the services ranked by the experts and those ranked by ELECTRE III and TOPSIS. Our analysis revealed that the KTD rankings produced by ELECTRE III outperformed those produced by TOPSIS.

Specifically, for the majority of the concepts (i.e., Temperature, Humidity, Wind_Speed, Drought_Factor, and Wind_Direction), the KTD measures for ELECTRE III were lower than those for TOPSIS, as shown in Figure 9. A decrease in the KTD measure indicates that the ranked lists produced by ELECTRE III are more similar to those proposed by the experts.

6.4 Experiment 3: Complexity Assessment of the Optimal Service Selection Using the Bα-DSS

The aim of this experiment is to evaluate the execution time of the Bα-DSS method for the selection of the optimal services compared with the Penalty-based Genetic Algorithm (GA) approach. The Genetic Algorithm (GA) generates a population of solutions, typically using random initialization, which are then evaluated based on a fitness function.

We employed the Penalty-based GA approach proposed in [24], which penalizes infeasible solutions that violate constraints. Table 3 outlines the parameter settings for the penalty-based GA, which were determined through experimentation on randomly generated test problems.

In our context, each chromosome in the GA represents an executable service composition. An executable service is formed by replacing each gene of the chromosome. Each gene in the chromosome corresponds to an index pointing to an array of potential concrete services that can fulfill a given concept.

We measured the computational time for the following approaches: the Penalty-based GA and the Bα-DSS. For all the tests, we used the six quality dimensions (QoS and QoDS), cited above

(in sub-Section 5.2, A). We varied the number of the candidate instance services for each concept.

For about 10 concrete services, the computation time of the Penalty-based GA tends to be linear, and almost constant. Then, we noticed, that the computational time for the Penalty-based GA increases exponentially as the number of service instances grows. Otherwise, the computational time in the GA rises exponentially, as the number of feasible candidate services and the number of concepts grow.

We, then, compared the computational time of the Bα-DSS approach with Penalty-based GA. We noticed that the computational time is narrowed significantly, as the number of service instances for a given concept increases, in the Bα-DSS approach. The use of the skyline operator and the α-dominance methods reduces the search space and saves time in the outranking process of the dominant solutions. Figure 11 depicts the necessary computational time for the Bα-DSS and the Penalty-based GA approaches.

6.5 Experiment 4: Scalability Assessement of the Optimal Service Selection Using the Bα-DSS

This experiment aims to evaluate the scalability of the Bα-DSS method for the selection of the optimal services, compared with the Penalty-based Genetic Algorithm (GA) approach. We varied a collection of the dataset of the service candidates for the selection process. This collection comprises datasets of varying sizes: 100K, 250K, 500K, 750K, and 1000K.

Figure 10 depicts the varied dataset size of the candidate services and the necessary computation time for the execution of the Bα-DSS and the Penalty-based GA. According to the results, we noticed that as the size of the dataset of the candidate services increases, the computational time decreases, with the application of our Bα-DSS approach.

Therefrom, the computation time decreases due to the pruning process of the candidate services that are not likely to be part of the optimal solutions of QoS services, thanks to the skyline and the α-dominance methods, which allowed to gain/save time on the selection process overall compared to the GA one.

Therefrom, the information overload issue related to the evolution of services and the need for context-specific selection, are addressed through our knowledge-driven solution (Bα-DSS) acting as a decision-maker and ensuring recommendations by filtering irrelevant services.

7 Threats to Validity

The final results of our proposal garnered significant attention from experts, as they have the potential to reduce considerably the initial set of services and the response-time, thanks to our (Bα-DSS) approach. Moreover, experts from the OSS conducted several evaluative tests, as described in Section 6, to evaluate the outcomes of our proposed automated knowledge-driven approach for optimal selection of services. In addition, using our framework, experts can apply weights based on the actual circumstances. The proposed framework is currently in a prototype stage, developed to meet the requirements specified by PREDICAT experts.

When evaluating the performance and quality of our framework, it is essential to consider the threats to the validity of the findings. Specifically, we need to assess the potential inaccuracies in the framework's outcomes, i.e., the relationship between the framework's results and reality. If the number of services significantly increases, the MESOn ontology may no longer provide adequate and timely responses for quality assessment. This could also affect the availability of information on service quality.

8 Conclusion and Future Work

This paper proposes a novel approach that combines (i) a dedicated ontology to define and assess data sources quality dimensions along with their associated inferences, with (ii) ELECTRE III MCDM method performing fuzzy outranking, to optimize the selection of services participating in service composition.

Additionally, our approach considers (iii) the knowledge related to the quality levels of both services (QoS) and environmental data sources (QoDS) in the outranking process.

To evaluate our framework, we conducted a series of experiments in collaboration with experts from OSS. Through these experiments, we assessed the effectiveness and relevance of our proposed approach. Our findings indicate that our framework offers:

1. Reasonable reasoning time for assessing data source quality, ensuring that data source quality cannot change within this timeframe.
2. Reduction of the execution time of the ELECTRE III method through the application of the skyline and the α-dominance methods. Furthermore, our results demonstrate that the ELECTRE III MCDM method outperforms the TOPSIS MCDM method in the ranking process and selection of optimal services.
3. A reduction of the computational time of the Bα-DSS approach compared with Penalty-based GA, as the number of service instances for a given concept increases.
4. The scalability analysis along with a variation of the dataset size of the services candidates showed a decrease of the computational time due to the pruning process of the irrelevant services.

As a future research, we intend to rely on the application of the reinforcement learning algorithms to select optimal candidates services. Furthermore, as part of the quality of services, we want to improve our framework with business non-functional qualities (e.g., consequences for variations, failure reporting, etc.), as future work.

However, since these QoS are not computable, we can rely on a subjective approach that allows evaluating these QoS, based on experts' ratings and feedback.

Acknowledgments

The authors acknowledge the support of the European Commission for funding the InnoRenew project (Grant Agreement #739574) under the Horizon 2020 Widespread-Teaming program, as well as the support of the Republic of Slovenia (Investment funding of the Republic of Slovenia and the European Regional Development Fund).

This work was financially supported by the "PHC Utique" program of the French Ministry of Foreign Affairs and Ministry of higher education and research and the Tunisian Ministry of higher education and scientific research in the CMCU project number 17G1122.

The authors acknowledge the European Commission for funding the InnoRenew CoE project (Grant Agreement #739574) under the Horizon2020 Widespread-Teaming program and the Republic of Slovenia (Investment funding of the Republic of Slovenia and the European Union of the European regional Development Fund).

References

1. **Abourezq, M., Idrissi, A. (2014).** Introduction of an outranking method in the cloud computing research and selection system based on the skyline. Proceedings of the IEEE 18th International Conference on Research Challenges in Information Science, pp. 1–12. DOI: 10.1109/RCIS.2014.6861067.
2. **Albertoni, R., Isaac, A. (2015).** Data quality vocabulary (DQV). W3C Interest Group Note. www.w3.org/TR/vocab-dqv/.
3. **Albertoni, R., Isaac, A. (2020).** Introducing the data quality vocabulary (DQV). Semantic Web, Vol. 12, No. 1, pp. 81–97. DOI: 10.3233/sw-200382.
4. **Alrifai, M., Skoutas, D., Risse, T. (2010).** Selecting skyline services for QoS-based web service composition. Proceedings of the 19th international conference on World wide web, pp. 11–20. DOI: 10.1145/1772690.1772693.

5. **Ardagna, D., Pernici, B. (2007).** Adaptive service composition in flexible processes. IEEE Transactions on software engineering, Vol. 33, No. 6, pp. 369–384.
6. **Balioti, V., Tzimopoulos, C., Evangelides, C. (2018).** Multi-criteria decision making using topsis method under fuzzy environment. Application in Spillway Selection. EWaS3 2018, Vol. 2, No. 11, pp. 637. DOI: 10.3390/proceedings2110637.
7. **Batini, C., Scannapieco, M. (2016).** Data and information quality: Dimensions, principles and techniques. DOI: 10.1007/978-3-319-24106-7.
8. **Ben-Ellefi, M., Bellahsene, Z., Breslin, J. G., Demidova, E., Dietze, S., Szymański, J., Todorov, K. (2018).** RDF dataset profiling – a survey of features, methods, vocabularies and applications. Semantic Web, Vol. 9, No. 5, pp. 677–705. DOI: 10.3233/sw-180294.
9. **Benouaret, K., Benslimane, D., Hadjali, A. (2011).** On the use of fuzzy dominance for computing service skyline based on QoS. Proceedings of the IEEE International Conference on Web Services, pp. 540–547. DOI: 10.1109/ICWS.2011.93.
10. **Benouaret, K., Benslimane, D., Hadjali, A., Barhamgi, M., Maamar, Z., Sheng, Q. Z. (2014).** Web service compositions with fuzzy preferences: A graded dominance relationship-based approach. ACM Transactions on Internet Technology, Vol. 13, No. 4, pp. 1–33. DOI: 10.1145/2576231.
11. **Bizer, C. (2007).** Quality-driven information filtering in the context of web-based information systems. Freie Universität Berlin. DOI: 10.17169/REFUBIUM-14260.
12. **Borzsony, S., Kossmann, D., Stocker, K. (2001).** The skyline operator. Proceedings of the 17th International Conference on Data Engineering, pp. 421–430. DOI: 10.1109/icde.2001.914855.
13. **Bosc, P., Pivert, O. (1995).** SQLf: A relational database language for fuzzy querying. IEEE Transactions on Fuzzy Systems, Vol. 3, No. 1, pp. 1–17. DOI: 10.1109/91.366566.
14. **Brans, J. P. (1982).** L'ingénierie de la décision: l'élaboration d'instruments d'aide a la décision. UniversitéLaval, Faculté des sciences de l'administration.
15. **Chen, L. (2014).** Ensuring reliability and QoS optimizing for web service composition. Proceedings of the 10th International Conference on Computational Intelligence and Security, pp. 510–513. DOI: 10.1109/cis.2014.86.
16. **Chouiref, Z., Belkhir, A., Benouaret, K., Hadjali, A. (2016).** A fuzzy framework for efficient user-centric web service selection. Applied Soft Computing, Vol. 41, pp. 51–65. DOI: 10.1016/j.asoc.2015.12.011.
17. **Dai, G., Zhu, Q. (2013).** Using skyline and dominance relationship for web services ranking. Journal of Computational Information Systems, Vol. 9, pp. 3977–3984. DOI: 10.12733/jcis5895.
18. **Debattista, J., Lange, C., Auer, S. (2014).** daQ, an ontology for dataset quality information. CEUR Workshop Proceedings, Vol. 1184, pp. 1–8.
19. **Dorfeshan, Y., Tavakkoli-Moghaddam, R., Jolai, F., Mousavi, S. M. (2021).** A new data-driven and knowledge-driven multi-criteria decision-making method. Journal of AI and Data Mining, Vol. 9, No. 4. DOI: 10.22044/jadm.2021.10803.2218.
20. **Farias-Lóscio, B., Stephan, E. G., Purohit, S. (2016).** Data on the web best practices: Dataset usage vocabulary. W3C Working Group Note. www.w3.org/TR/vocab-duv/.
21. **Fielding, R. T. (2000).** Architectural styles and the design of network-based software architectures. University of California, Irvine.
22. **Forestal, R. L., Pi, S. (2021).** A hybrid approach based on ELECTRE III-genetic algorithm and TOPSIS method for selection of optimal COVID-19 vaccines. Journal of

Multi-Criteria Decision Analysis, Vol. 29, No. 1-2, pp. 80–91. DOI: 10.1002/mcda.1772.

23. **Frank, M., Walker, J. (2016).** User centred methods for measuring the value of open data. The Journal of Community Informatics, Vol. 12, No. 2. DOI: 10.15353/joci.v12i2.3221.
24. **Grati, R., Boukadi, K., Ben-Abdallah, H. (2014).** QoS based resource allocation and service selection in the cloud. Proceedings of the 11th International Conference on e-Business, pp. 249–256.
25. **Jarke, M., Jeusfeld, M. A., Quix, C., Vassiliadis, P. (1999).** Architecture and quality in data warehouses: An extended repository approach. Information Systems, Vol. 24, No. 3, pp. 229–253. DOI: 10.1016/s0306-4379(99)00017-4.
26. **Jauhari, A., Mufarroha, F. A. (2020).** Smart mobile application for decision support systems on determination of resident in dormitory. Jurnal Ilmiah Kursor, Vol. 10, No. 3. DOI: 10.21107/kursor.v10i3.236.
27. **Jayawardene, V., Sadiq, S., Indulska, M. (2015).** An analysis of data quality dimensions. Technical Report, The University of Queensland.
28. **Kaufer, F., Klusch, M. (2006).** WSMO-MX: A logic programming based hybrid service matchmaker. Proceedings of the European Conference on Web Services, pp. 161–170. DOI: 10.1109/ECOWS.2006.39.
29. **Klusch, M., Fries, B., Sycara, K. (2009).** OWLS-MX: A hybrid semantic web service matchmaker for OWL-S services. Journal of Web Semantics, Vol. 7, No. 2, pp. 121–133. DOI: 10.1016/j.websem.2008.10.001.
30. **Koppen, M., Vicente-Garcia, R. (2004).** A fuzzy scheme for the ranking of multivariate data and its application. Proceedings of the IEEE Annual Meeting of the Fuzzy Information, Vol. 1, pp. 140–145. DOI: 10.1109/NAFIPS.2004.1336266.
31. **Kumar, R. R., Kumari, B., Kumar, C. (2020).** CCS-OSSR: A framework based on hybrid MCDM for optimal service selection and ranking of cloud computing services. Cluster Computing, Vol. 24, No. 2, pp. 867–883. DOI: 10.1007/s10586-020-03166-3.
32. **Lebo, T., Sahoo, S., McGuinness, D. (2012).** PROV-O: The PROV ontology. W3C Recommendation. www.w3.org/TR/prov-o/.
33. **Liangzhao, Z., Benatallah, B., Ngu, A. H. H., Dumas, M., Kalagnanam, J., Chang, H. (2004).** QoS-aware middleware for web services composition. IEEE Transactions on Software Engineering, Vol. 30, No. 5, pp. 311–327. DOI: 10.1109/TSE.2004.11.
34. **Maali, F., Erickson, J., Archer, P. (2014).** Data catalog vocabulary (DCAT). W3C recommendation. www.w3.org/TR/vocab-dcat-3/.
35. **Olson, D. L. (2004).** Comparison of weights in TOPSIS models. Mathematical and Computer Modelling, Vol. 40, No. 7-8, pp. 721–727. DOI: 10.1016/j.mcm.2004.10.003.
36. **Ouadah, A., Benouaret, K., Hadjali, A., Nader, F. (2015).** Combining skyline and multi-criteria decision methods to enhance web services selection. Proceedings of the 12th International Symposium on Programming and Systems (ISPS), pp. 1–8. DOI: 10.1109/ISPS.2015.7244975.
37. **Ouadah, A., Benouaret, K., Hadjali, A., Nader, F. (2015).** SkyAP-S3: A hybrid approach for efficient skyline services selection. Proceedings of the IEEE 8th International Conference on Service-Oriented Computing and Applications, pp. 18–25. DOI: 10.1109/SOCA.2015.22.
38. **Ouadah, A., Hadjali, A., Nader, F. (2018).** A hybrid MCDM framework for efficient web services selection based on QoS. Proceedings of the International Conference on Applied Smart Systems, pp. 1–6. DOI: 10.1109/icass.2018.8652037.
39. **Papadias, D., Tao, Y., Fu, G., Seeger, B. (2003).** An optimal and progressive algorithm for skyline queries. Proceedings of the

ACM SIGMOD International Conference on Management of Data, pp. 467–478. DOI: 10 .1145/872757.872814.

40. **Pipino, L. L., Lee, Y. W., Wang, R. Y. (2002).** Data quality assessment. Communications of the ACM, Vol. 45, No. 4, pp. 211–218. DOI: 10 .1145/505248.506010.
41. **Polska, O. V., Kudermetov, R. K., Shkarupylo, V. V. (2021).** An approach web service selection by quality criteria based on sensitivity analysis of MCDM methods. Radio Electronics, Computer Science, Control, , No. 2, pp. 133–143. DOI: 10.15588/1607-3274-2021-2-14.
42. **Rodriguez-Mier, P., Pedrinaci, C., Lama, M., Mucientes, M. (2016).** An integrated semantic web service discovery and composition framework. IEEE Transactions on Services Computing, Vol. 9, No. 4, pp. 537–550. DOI: 10.1109/tsc.2015.2402679.
43. **Roy, B. (1991).** The outranking approach and the foundations of electre methods. Theory and Decision, Vol. 31, No. 1, pp. 49–73. DOI: 10.1007/bf00134132.
44. **Serrai, W., Abdelli, A., Mokdad, L., Hammal, Y. (2017).** Towards an efficient and a more accurate web service selection using MCDM methods. Journal of Computational Science, Vol. 22, pp. 253–267. DOI: 10.1016/j.jocs.2 017.05.024.
45. **Serrai, W., Abdelli, A., Mokdad, L., Serrai, A. (2018).** How to deal with QoS value constraints in MCDM based web service selection. Concurrency and Computation: Practice and Experience, Vol. 31, No. 24. DOI: 10.1002/cp e.4512.
46. **Sun, L., Ma, J., Zhang, Y., Dong, H., Hussain, F. K. (2016).** Cloud-FuSeR: Fuzzy ontology and MCDM based cloud service selection. Future Generation Computer Systems, Vol. 57, pp. 42–55. DOI: 10.1016/j.future.2015.11.025.
47. **Tan, K. L., Eng, P. K., Ooi, B. C. (2001).** Efficient progressive skyline computation. Proceedings of the 27th International Conference on Very Large Data Bases, pp. 301–310.
48. **Wang, R. Y., Strong, D. M. (1996).** Beyond accuracy: What data quality means to data consumers. Journal of Management Information Systems, Vol. 12, No. 4, pp. 5–33. DOI: 10.1080/07421222.1996.11518099.
49. **Wang, X., Vitvar, T., Kerrigan, M., Toma, I. (2006).** A QoS-aware selection model for semantic web services. Lecture Notes in Computer Science, pp. 390–401. DOI: 10.100 7/11948148_32.
50. **Wauthier, F., Jordan, M., Jojic, N. (2013).** Efficient ranking from pairwise comparisons. Proceedings of the 30th International Conference on Machine Learning, Vol. 28, No. 3, pp. 109–117.
51. **Xu, J., Guo, L., Zhang, R., Zhang, Y., Hu, H., Wang, F., Pei, Z. (2017).** Towards fuzzy QoS driven service selection with user requirements. Proceedings of the International Conference on Progress in Informatics and Computing, pp. 230–234. DOI: 10.1109/PIC. 2017.8359548.
52. **Youssef, A. E. (2020).** An integrated MCDM approach for cloud service selection based on TOPSIS and BWM. IEEE Access, Vol. 8, pp. 71851–71865. DOI: 10.1109/ACCESS.2 020.2987111.
53. **Yu, Q., Bouguettaya, A. (2010).** Computing service skylines over sets of services. Proceedings of the IEEE International Conference on Web Services, pp. 481–488. DOI: 10.1109/ICWS.2010.48.
54. **Yu, Q., Bouguettaya, A. (2010).** Foundations for Efficient Web Service Selection. Springer US. DOI: 10.1007/978-1-4419-0314-3.
55. **Zadeh, L. A. (1965).** Fuzzy sets. Information and Control, Vol. 8, No. 3, pp. 338–353. DOI: 10.1016/s0019-9958(65)90241-x.
56. **Zou, H., Zhang, L., Yang, F., Zhao, Y. (2010).** A web service composition algorithmic method based on topsis supporting multiple

decision-makers. Proceedings of the 6th World Congress on Services, pp. 158–159. DOI: 10 .1109/services.2010.110.

57. **Zulqarnain, R., Saeed, M., Ahmad, N., Dayan, F., Ahmad, B. (2020).** Application of topsis method for decision making. International Journal of Scientific Research in Mathematical and Statistical Sciences, Vol. 7, No. 2, pp. 76–81.

Article received on 07/01/2023; accepted on 03/03/2024.
** Corresponding author is Hela Taktak.*

ISSN 2007-9737

On the Performance Assessment and Comparison of Features Selection Approaches

Seyyid Ahmed Medjahed[1,*], Fatima Boukhatem[2]

[1] University of Relizane,
Algeria

[2] University Djillali Liabes,
Algeria

seyyidahmed.medjahed@univ-relizane.dz, fatima.boukhatem@univ-sba.dz

Abstract. In many supervised learning problems, feature selection techniques are increasingly essential across various applications. Feature selection significantly influences the classification accuracy rate and the quality of SVM model by reducing the number of features, remove irrelevant and redundant features. In this paper, we evaluate the performance of twenty feature selection algorithms over four databases. The performance is conducted in term of: classification accuracy rate, Kuncheva's Stability, Information Stability, SS Stability and SH Stability. To measure the feature selection algorithms, multiple datasets from the UCI Machine Learning Repository are utilized to assess both classification accuracy and stability variations.

Keywords. Feature selection, classification, stability, support vector machine.

1 Introduction

In recent years, the motivation behind applying feature selection techniques has evolved significantly. What was once merely an illustrative example has now become a crucial prerequisite for effective model building. This shift in emphasis can be attributed to several factors, including improved generalization performance, reduced running time requirements, and the need to address constraints and interpretational challenges inherent in the problem domain.

Feature selection is a vital dimensionality reduction technique in data mining, involving the selection of a subset of original features based on specific criteria.

This process is important and commonly utilized to enhance the efficiency and effectiveness of data analysis tasks [1, 2, 3].

It reduces the number of features, removes irrelevant, redundant, or noisy data, and brings the immediate effects for applications: speeding up a data mining algorithm, and improving mining performance such as predictive accuracy and result comprehensibility.

Therefore, it is essential to employ an effective feature selection method that considers the number of features used for sample classification to enhance processing speed, predictive accuracy, and comprehensibility.

The correlation between features significantly impacts classification outcomes. Removing important features can reduce classification accuracy and negatively affect the quality of SVM models.

Similarly, certain features may have no discernible effect or may be laden with high levels of noise [4]. Their removal increases the classification accuracy rate.

The aim of feature selection is to find the smallest feature subset that increases the classification accuracy rate.

The optimal features subset is not unique; it may be possible to achieve the same accuracy rate using different sets of features, because if two features are correlated one can replace by other.

Note that feature subset selection chooses a set of features from existing features, and does not construct new ones; there is no feature extraction or construction [5, 6].

Table 1. Some feature selection criteria and algorithms

Methods	Full Name
MRMR	Max-Relevance Min-Redundancy [20,18]
CMIM	Conditional Mutual Info Maximisation [13,18]
JMI	Joint Mutual Information [14,18]
DISR	Double Input Symmetrical Relevance [15,18]
CIFE	Conditional Infomax Feature Extraction [16,18]
ICAP	Interaction Capping [17,18]
CONDRED	Conditional Redundancy [18]
BETAGAMMA	BetaGamma [18]
MIFS	Mutual Information Feature Selection [19,18]
CMI	Conditional Mutual Information [18]
MIM	Mutual Information Maximisation [12,18]
RELIEF	Relief [18]
FCBF	Fast Correlation Based Filter [21,27]
MRF	Markov Random Fields [26]
SPEC	Spectral [22,27]
T-TEST	Student's T-test [27]
KRUSKAL-WALLIS	Kruskal-Wallis Test [23,27]
FISHER	Fisher Score [24,27]
GINI	Gini Index [25,27]
GA	Genetic Algorithm

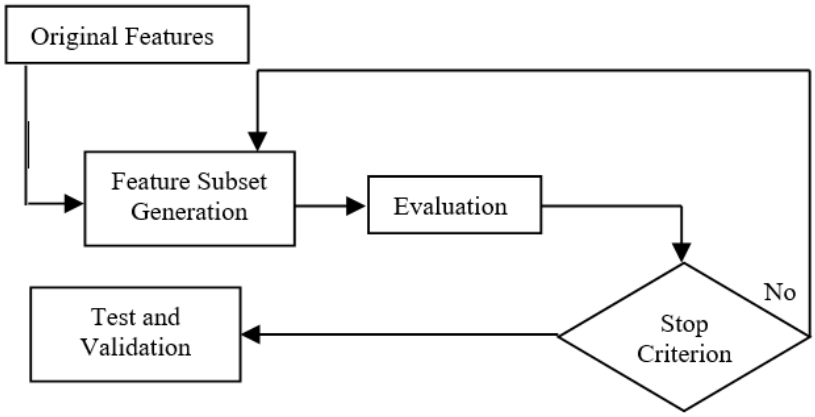


Fig. 1. A unified view of feature selection process

In this study, we analyze and evaluate the performance of several feature selection techniques (20 algorithms) by using the criterion of stability and the classification accuracy rate calculates with SVM-SMO.

The experimentation is conducted over 4 datasets obtained from UCI machine learning repository.

The paper is organized as follows. In section 2, we give an overview of SVM.

Table 2. Datasets from the UCI ML repository

Datasets	Number of classes	Number of instances	Number of features
Breast Cancer	2	699	9
Cardiotocography	2	1831	21
ILPD	2	583	9
Mammographic Mass	2	961	5

Table 3. Number of instances used for training and testing steps

Datasets	Missing instances	Training set	Testing set
Breast Cancer	16	411	272
Cardiotocography	0	1101	730
ILPD	0	351	232
Mammographic Mass	131	500	330

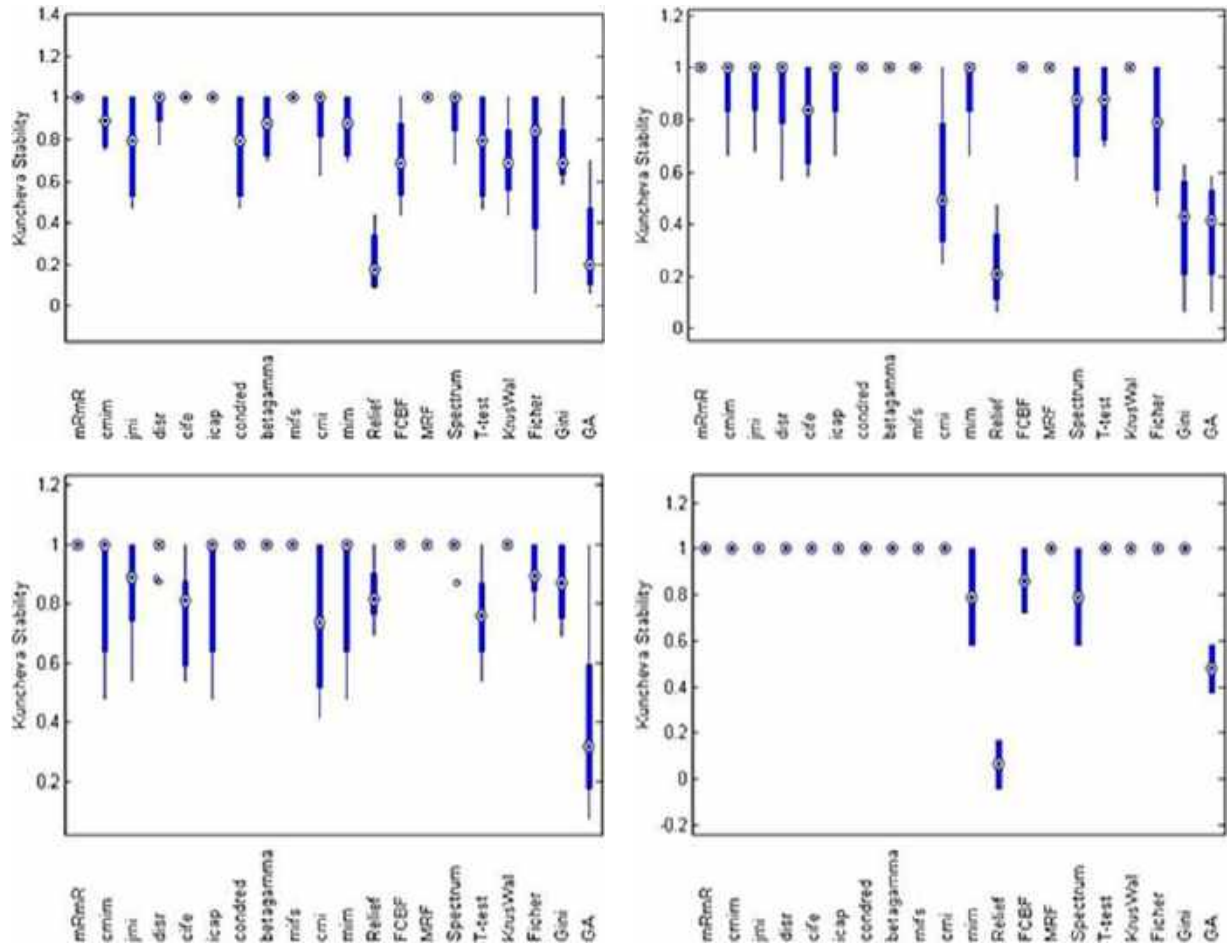


Fig. 2. Kuncheva's stability over the 4 data sets. The box indicates the upper and the lower quartiles. The small circle shows the median values, while the blue line indicates the maximum and the minimum values

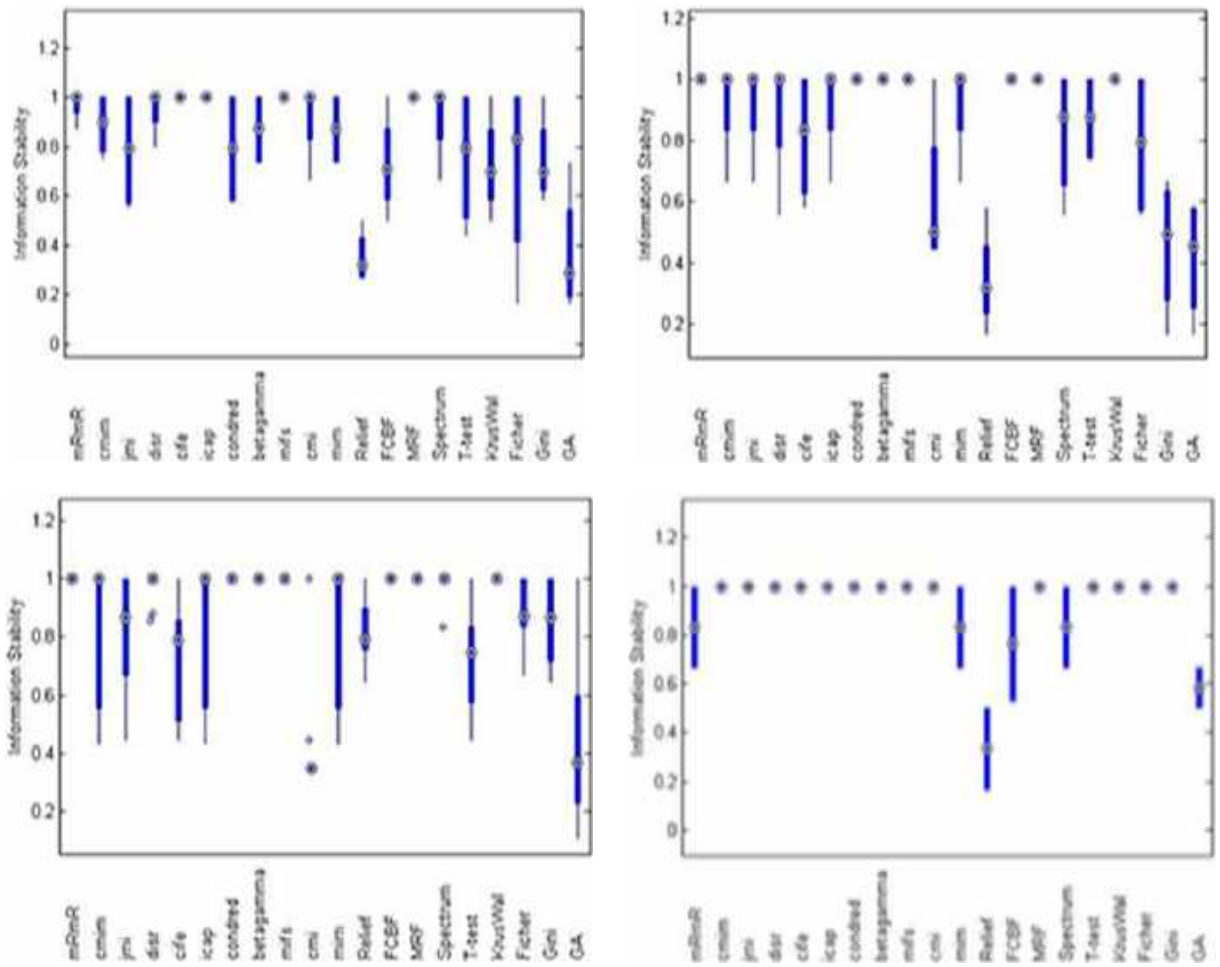


Fig. 3. Information stability over the 4 data sets. The box indicates the upper and the lower quartiles. The small circle shows the median values, while the blue line indicates the maximum and the minimum values

In section 3, we describe the stability criteria used in the literature. In section 4, we discuss the different feature selection techniques.

Section 5 describes the results obtained by the approaches. Finally, concluding remarks are made in section 6.

2 Overview of Support Vector Machine

SVM can be briefly described as follows [7, 8, 9]. Consider $(x_1, y_1), \cdots, (x_n, y_n)$ with $y\{-1, +1\}$ denote a set of training data. The goal of Support Vector Machines (SVM) is to create a separating hyperplane in the attribute space that maximizes the margin between instances of different classes. This task involves reformulating the classification problem into a quadratic optimization problem aimed at finding the optimal hyperplane:

$$\begin{aligned} &\min_{\alpha} -\sum_{i=1}^{N} \alpha_i + \frac{1}{2}\sum_{i,j} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle, \\ &\text{s. c.} \sum_{i=1}^{N} \alpha_i y_i = 0, \\ &\forall i \in \{1, \ldots, N\}, \qquad \alpha_i \geq 0. \end{aligned} \tag{1}$$

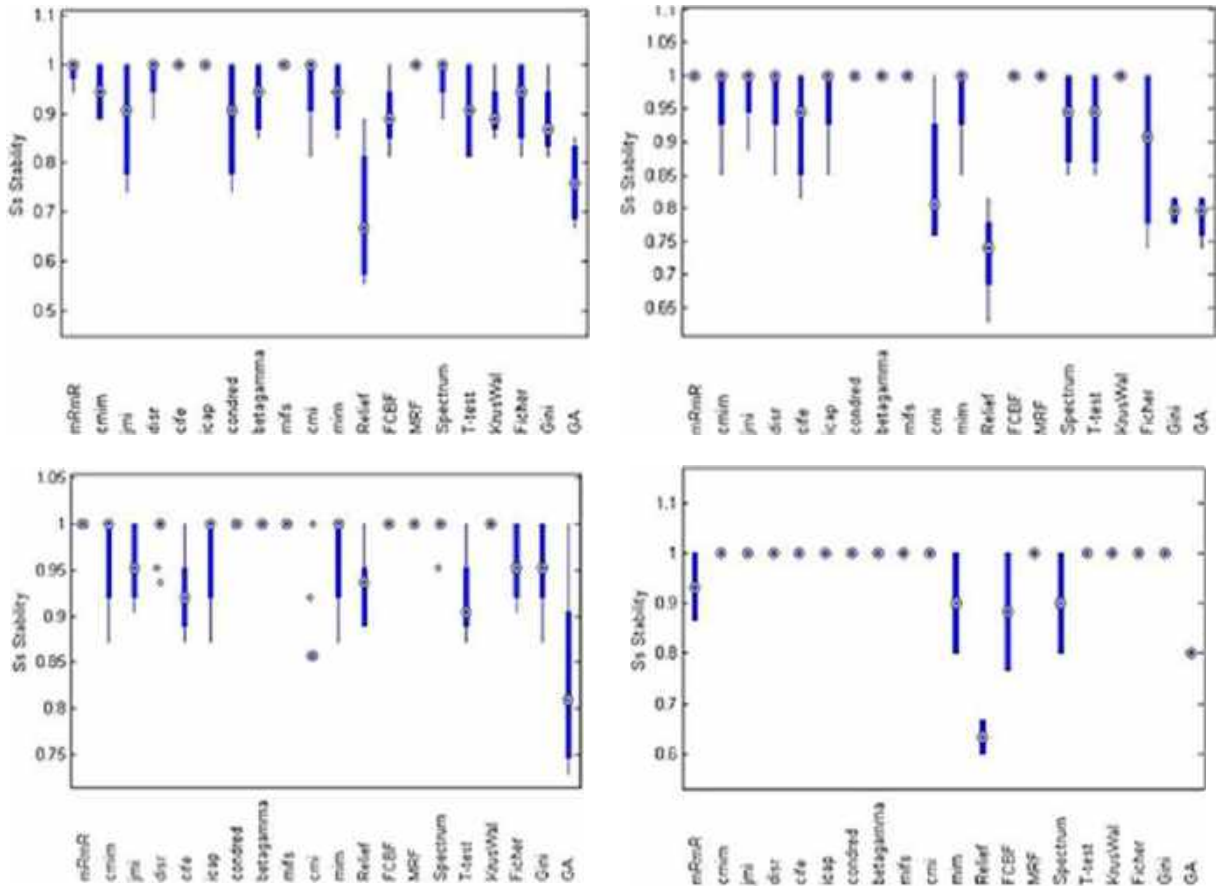


Fig. 4. SS stability over the 4 data sets. The box indicates the upper and the lower quartiles. The small circle shows the median values, while the blue line indicates the maximum and the minimum values

This is the dual form of the quadratic problem, C represents the regularization parameter. To solve the optimization problem in Support Vector Machines (SVM), quadratic optimization algorithms are utilized.

Some commonly used algorithms include: Sequential Minimal Optimization [10, 11], Trust Region, etc. By solving the optimization problem, we determine the Lagrange multipliers, the optimal hyperplane is given by:

$$w^* = \sum_{i=1}^{N} \alpha_i y_i x_i,$$
$$\mathrm{b}^* = -\frac{1}{2}\langle w^*, x_r + x_s \rangle, \quad (2)$$
$$H(x) = \mathrm{sign}(\langle \mathrm{w}^*, \mathrm{x} \rangle + \mathrm{b}^*),$$

where $\alpha_r, \alpha_s > 0$, $y_r = -1$, $y_s = 1$.

3 Feature Selection Algorithm

Feature selection is a domain garnering growing attention within the realm of machine learning. Numerous feature selection techniques have been outlined in literature dating back to the 1970s.

Feature selection algorithms are categorized into three main types based on their strategies: filter, wrapper, and embedded models.

Filter feature selection methods do not consider classifier properties; instead, they conduct statistical tests on variables. In contrast, wrapper feature selection evaluates various feature sets by constructing classifiers.

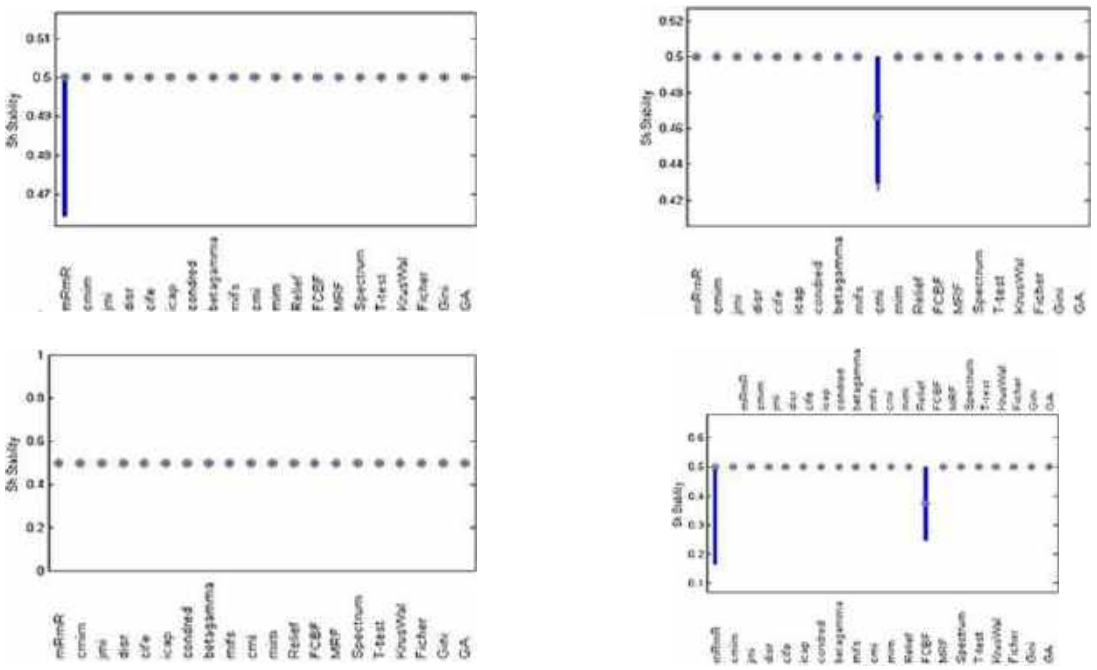


Fig. 5. S_H stability over the 4 data sets. The box indicates the upper and the lower quartiles. The small circle shows the median values, while the blue line indicates the maximum and the minimum values

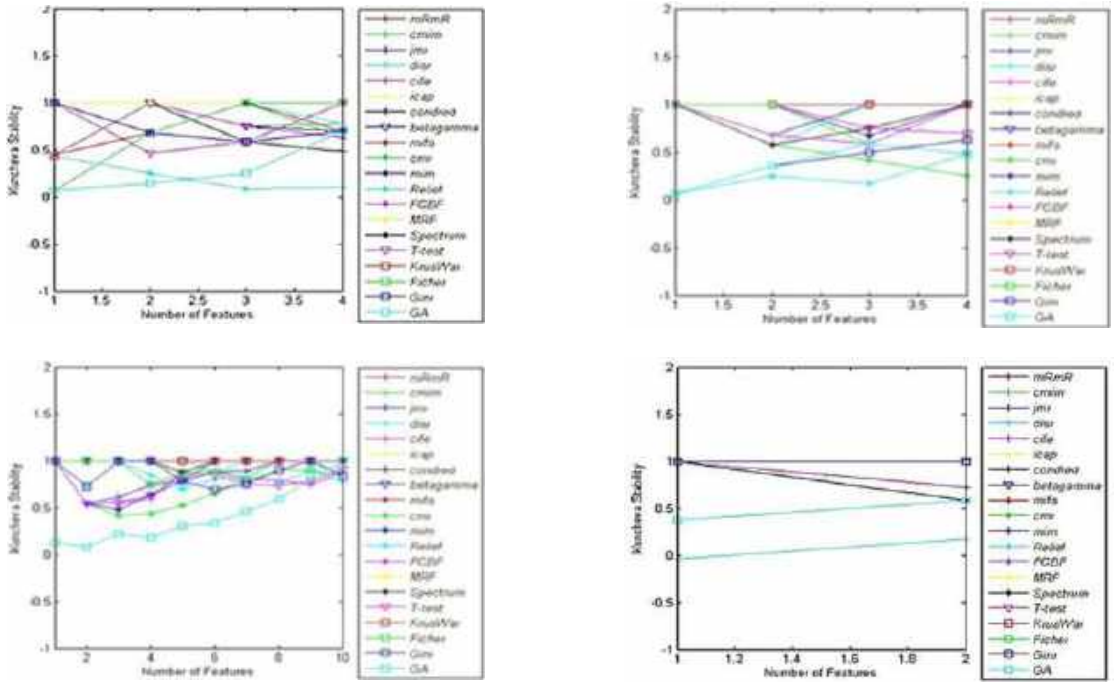


Fig. 6. Kuncheva's stability over the 4 data sets for each number of selected features

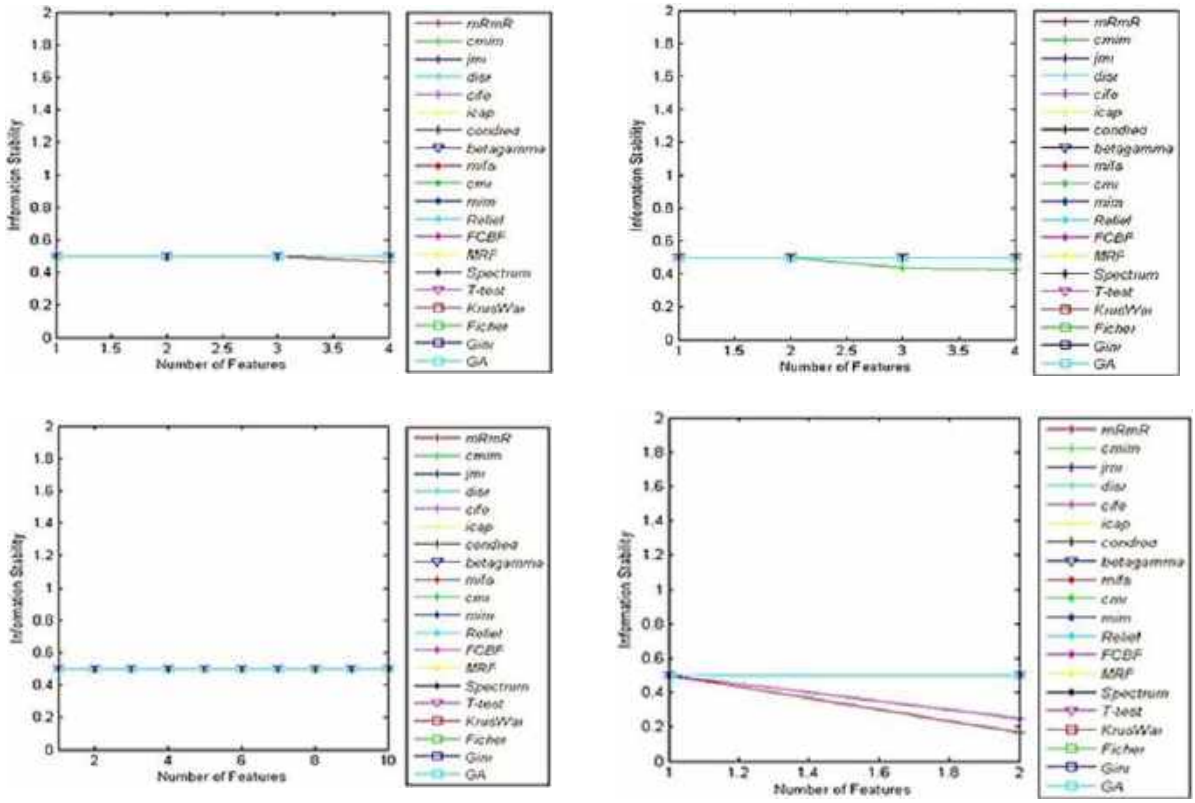


Fig. 7. Information stability over the 4 data sets for each number of selected features

Embedded model algorithms integrate variable selection into the training process, deriving feature relevance analytically from the learning model's objective. Table 1 summarizes some feature selection criteria and algorithms.

4 Stability of Feature Selection Algorithm

The stability of a feature selection algorithm refers to how sensitive it is to changes in feature preferences or rankings. It quantifies how different training set affect the feature preferences [31]. To calculate the stability, we require a similarity measure for feature preferences: Consider two subsets A and B we denote:

| . | The cardinality.

∩ The union.

U The intersection.

4.1 SS Stability

Kalousis et al. [29] define the similarity index between two subsets, A and B, as:

$$s_s = 1 - \frac{|A| + |B| - 2|A \cap B|}{|A| + |B| - |A \cap B|} = \frac{|A \cap B|}{|A \cup B|}. \quad (3)$$

The SS stability is a simple adaptation of the Tanimoto, which measures the similarity distance between two sets A and B. SS takes values in [0,1] with 0 meaning that there is no overlap between the two sets, and 1 that the two sets are identical.

4.2 SH Stability

Dunne et al. [30] calculates the similarity between two subsets by comparing the relative Hamming distance of their corresponding masks. In set notation, this method can be described as follows:

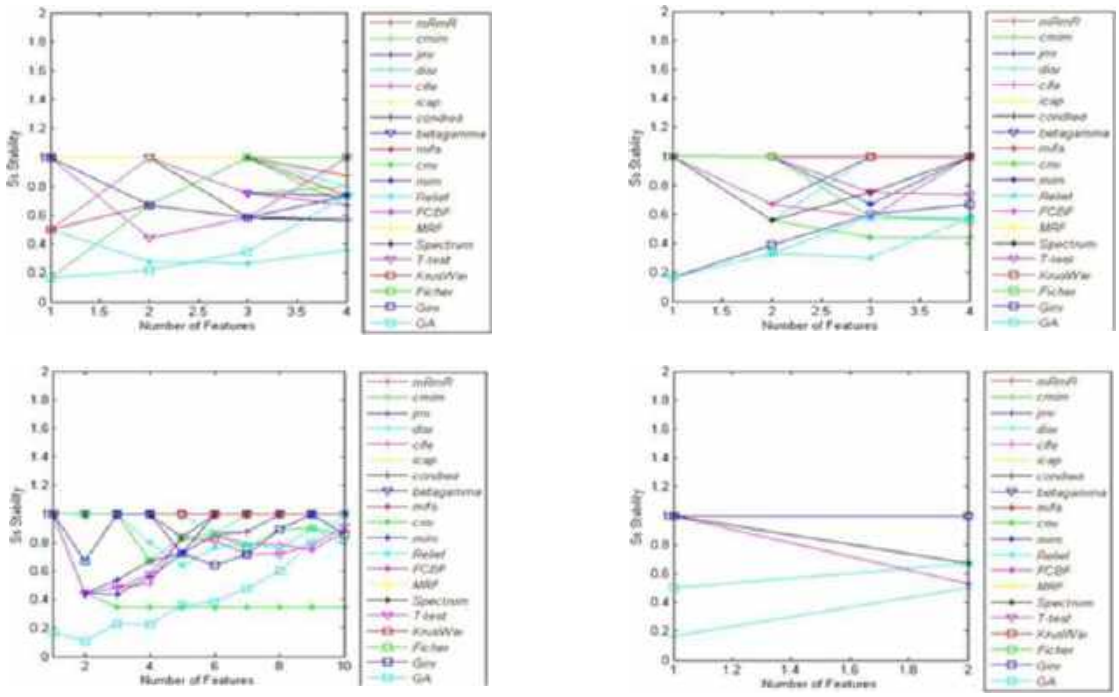


Fig. 8. SS stability over the 4 data sets for each number of selected features

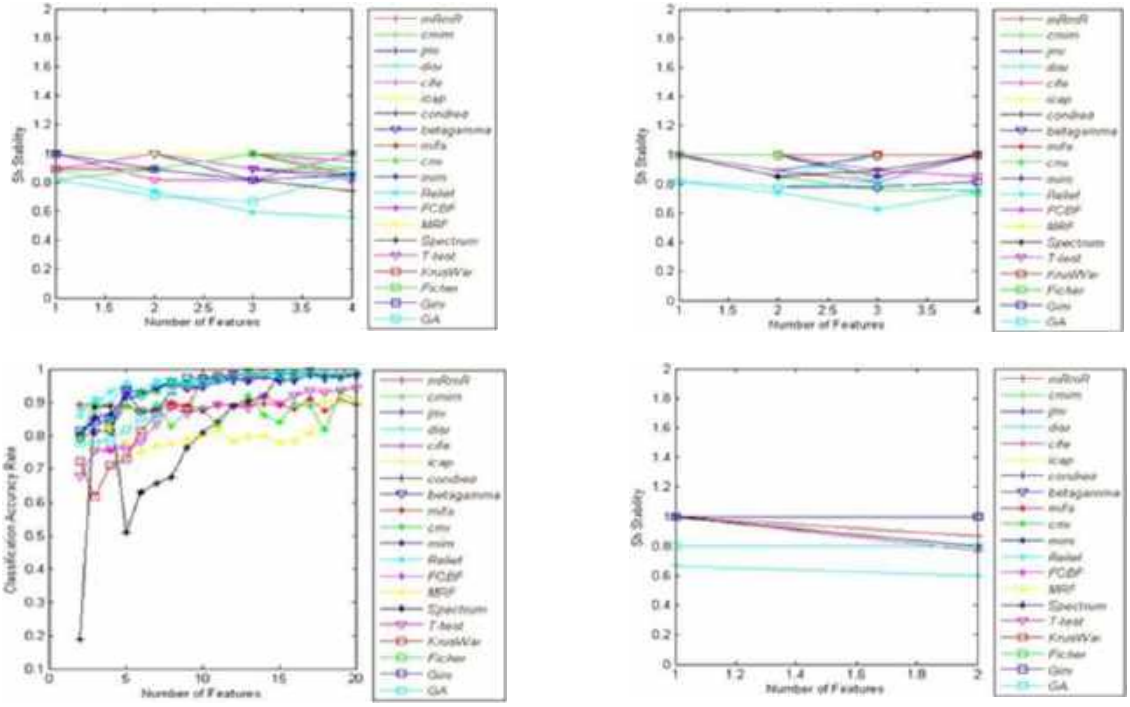


Fig. 9. SH stability over the 4 data sets for each number of selected features

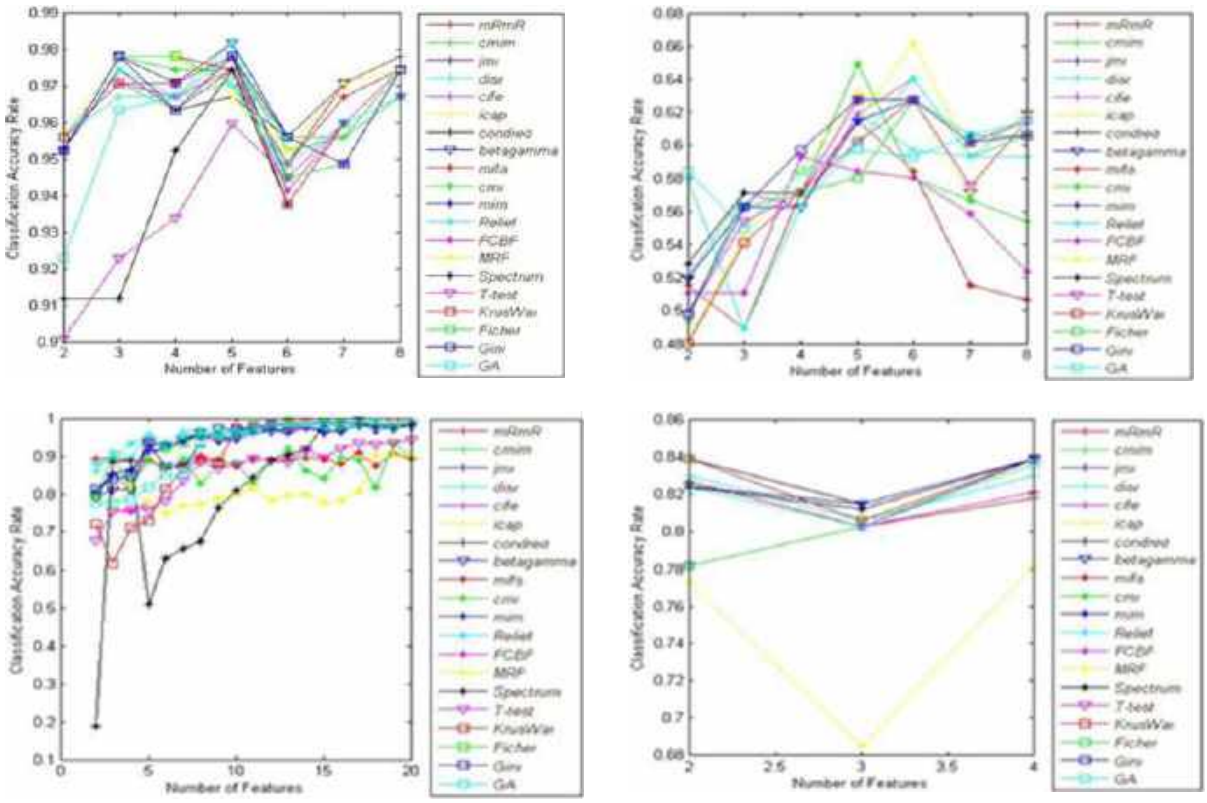


Fig. 10. The classification accuracy rate for each number of features

$$s_H = 1 - \frac{|A \setminus B| + |B \setminus A|}{n}. \quad (4)$$

4.3 Kuncheva Stability

Kuncheva [32] define the consistency index for two subsets with the same cardinality as:

$$I_c = \frac{r - \frac{k^2}{n}}{k - \frac{k^2}{n}} = \frac{rn - k^2}{k(n-k)}, \quad (5)$$

where $k = |A| = |B|$ and $r = |A \cap B|$. The maximum value of the index is $IC = 1$.it mean that $r = k$, and the minimum value is $IC = -1$.

4.4 Information Stability

Lei Yu et al. [33, 34] propose the normalized mutual information as a measure of stability of two feature sets:

$$\mathrm{Sim}(x_a, x_b) = \frac{I(x_a, x_b)}{H(x_a) + H(x_b)}. \quad (6)$$

The stability of a set of sequences features, $F = \{S_1, S_2, \ldots, S_K\}$ is the average of all pairwise.

5 Experimental Results

In this section, we have made a comparison protocol between the several feature selections techniques defined in the literature and shown the performance of each technique.

The experiment is analyzed by using the following performance measures: classification accuracy rate calculated by using the support vector machine. Also, we use the stability criteria: Kuncheva stability, Information stability, SS and SH stability. Table 2 presents a summary of four selected datasets used in the feature selection experiment:

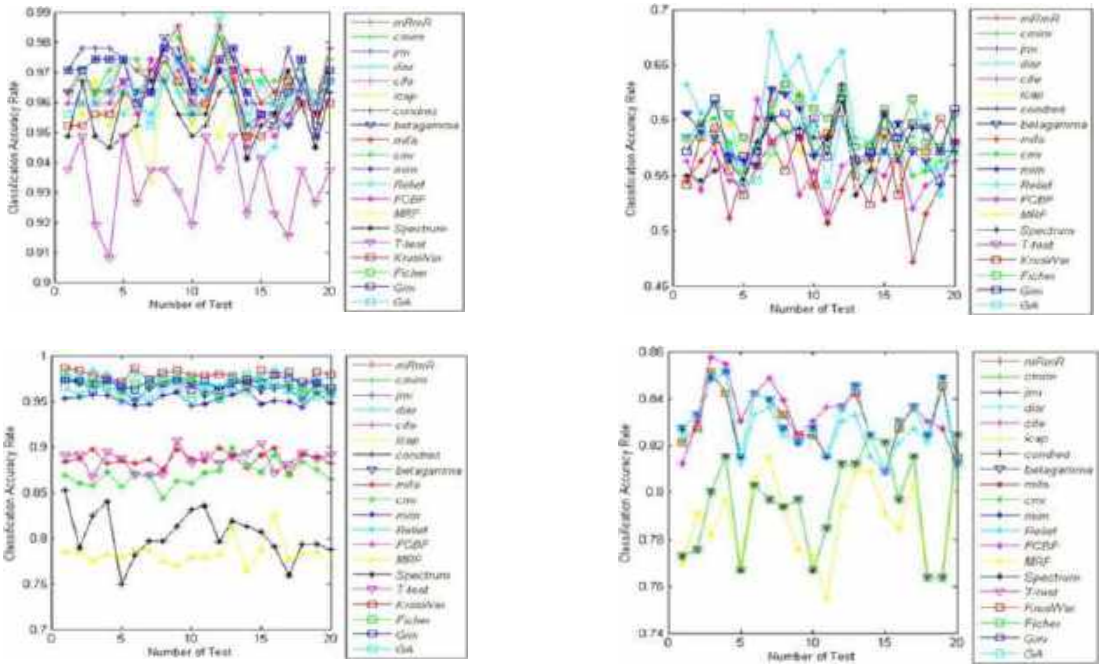


Fig. 11. The classification accuracy rate for each training set over the 4 better features

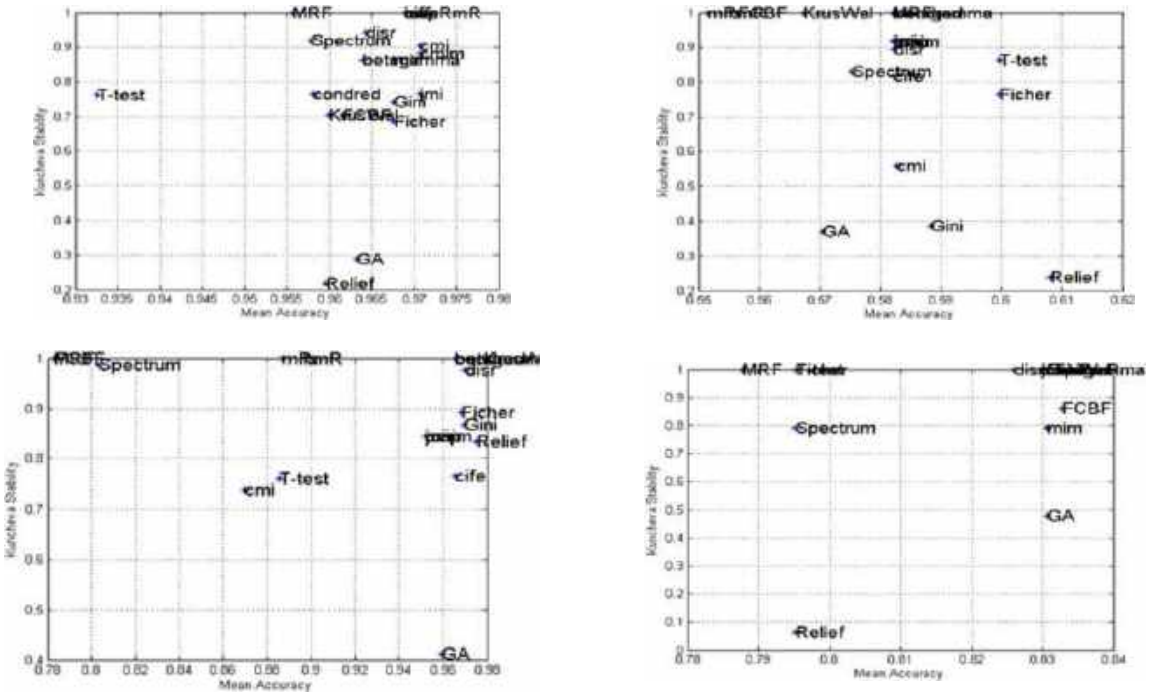


Fig. 12. Kuncheva's stability versus the average classification accuracy rate over 20 different training sets for each dataset

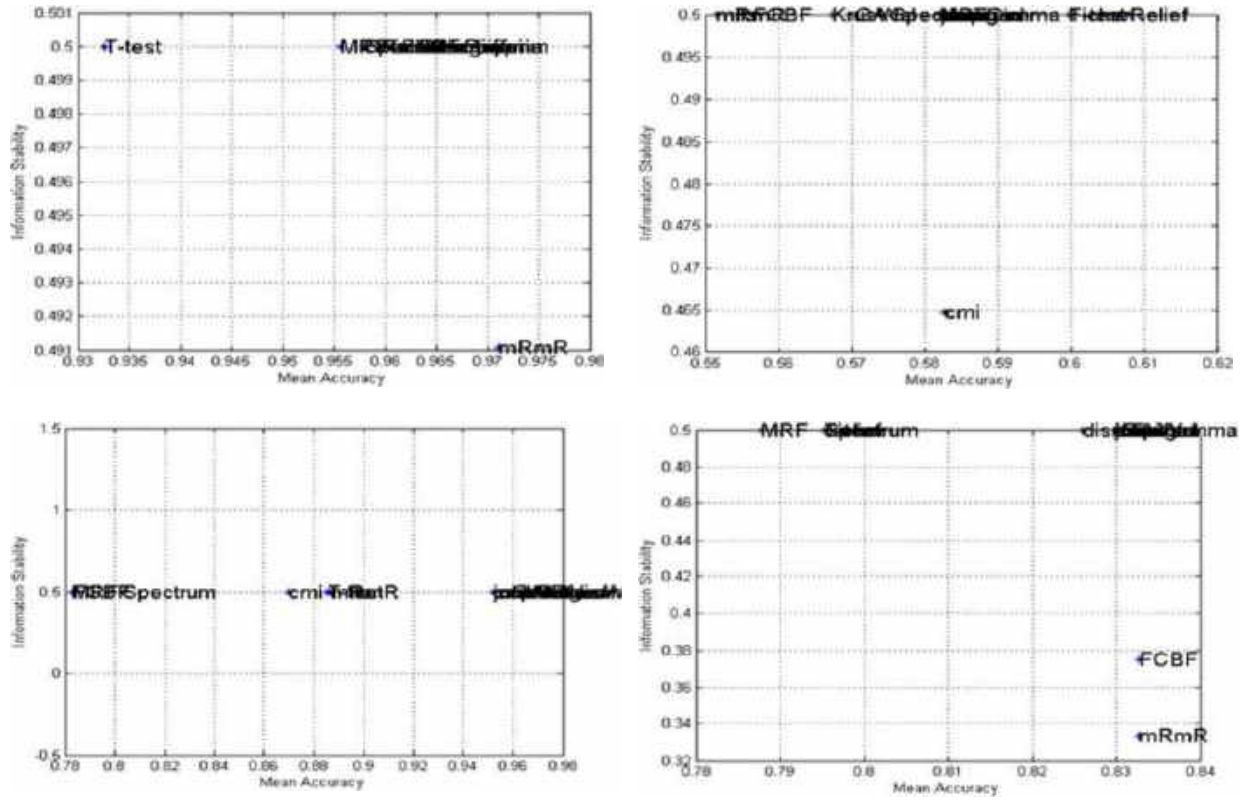


Fig.13. Information stability versus the average classification accuracy rate over 20 different training sets for each data

WDBC (Wisconsin Dataset Breast Cancer), Cardiotocography, ILPD (Indian Liver Patient Dataset), and Mammographic Mass. The performance evaluation of feature selection techniques requires the determination of the training and testing set.

In this study, we split randomly the initial dataset by using the hold out method which is a king of cross validation. In this experiment, less than one-third of the initial data is allocated for testing purposes. Specifically, 60% of the instances are designated for training, while the remaining 40% are reserved for testing.

Table 3 outlines the number of instances utilized during both the training and testing phases for each dataset. To compare the feature selection criteria defined above, we proceed as follows: for each data set, we select different training set and we take a set of features for each training set by using each feature selection criterion.

The following figures 2,3,4,5 show the Kuncheva's Stability, Information Stability, SS Stability and SH Stability measures over 4 datasets for each feature selection criterion. For each data set we calculate the stability for different training set obtained by using the hold out method which selects randomly a training set.

10 training sets are selected for each data set, we use this principle to better exploit each dataset. The results show that for all the training set which are selected randomly for each data sets, all the methods are stable except GA, CMI, T-test, Fisher, Gini, and relief.

The stability for JMI, MRMR, Disr, Condred, Mifs, FCBF, MRF and Kruskal-Wallis is equal to 1 for all the datasets, this means that these methods have select the same subset of feature for each training set of the four datasets. Therefore, theses feature selection criterions have selected the relevant subset of feature.

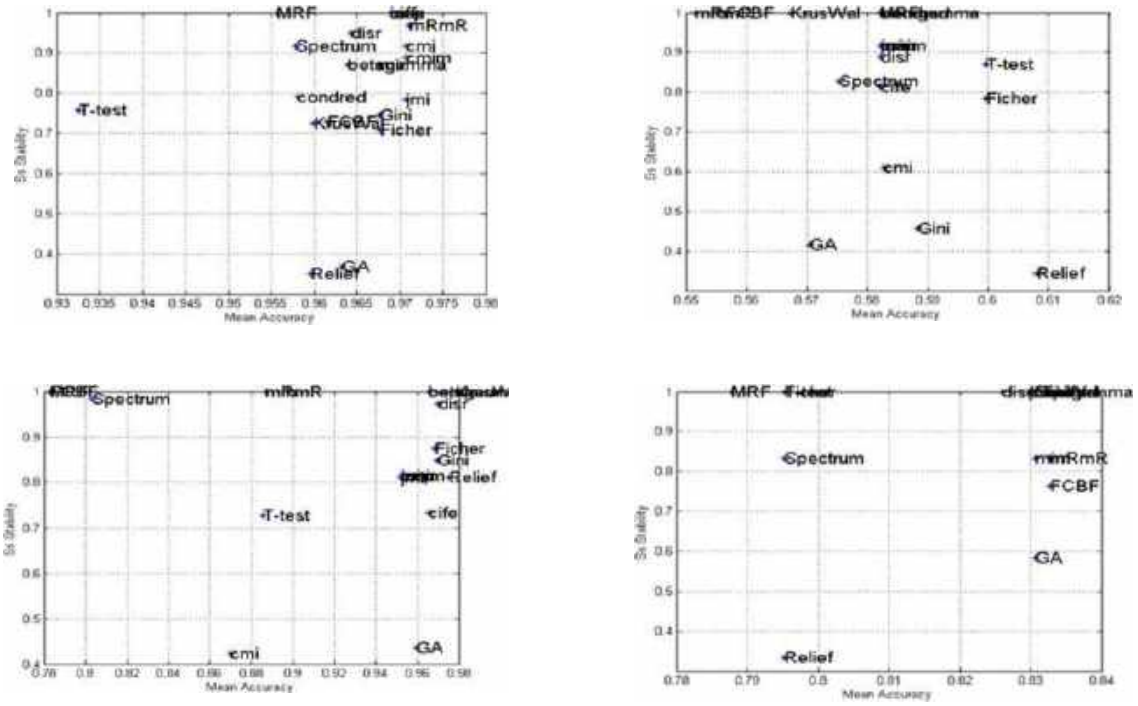


Fig. 14. SS stability versus the average classification accuracy rate over 20 different training sets for each data set

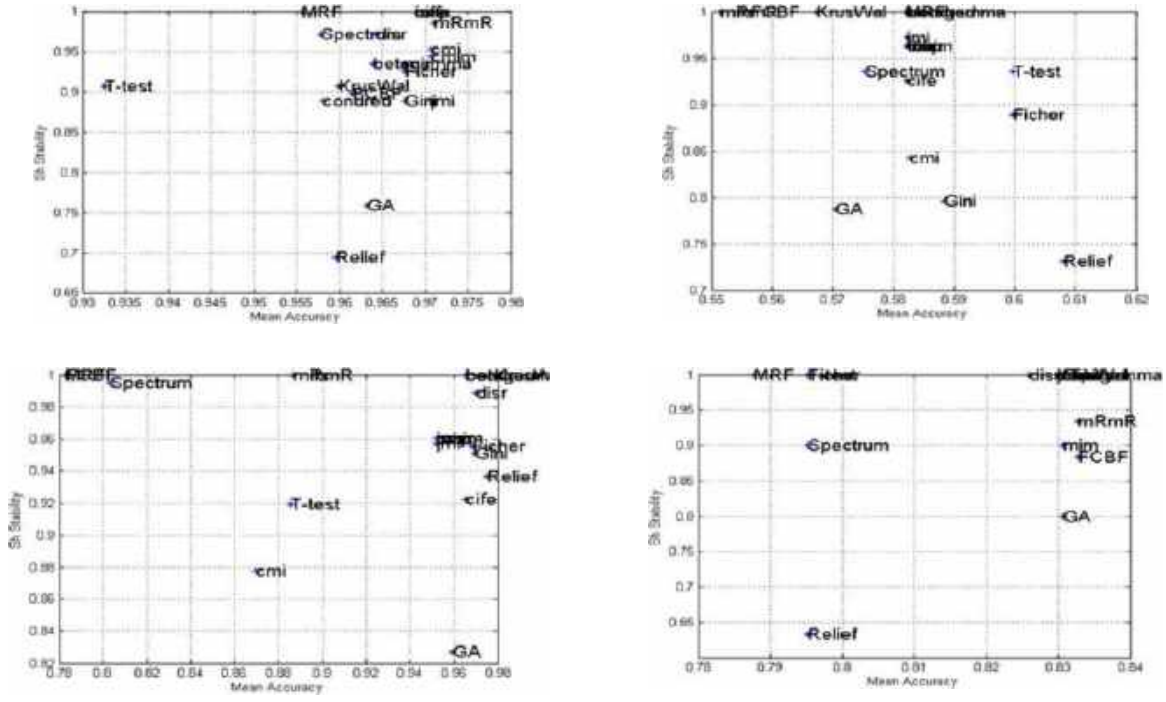


Fig. 15. SH stability versus the average classification accuracy rate over 20 different training sets for each dataset

Table 4. Average classification accuracy rate for each data set. Filled cell represents the higher accuracy rate

Methods	Average classification accuracy rate (%)			
	WDBC	**ILPD**	**Cardio**	**Mammo**
MRMR	97,11	55,15	88,68	83,27
CMIM	97,07	58,24	95,21	82,98
JMI	97,07	58,22	95,21	82,98
DISR	96,43	58,22	97,00	82,60
CIFE	96,89	58,22	96,55	82,98
ICAP	96,89	58,24	95,21	82,98
CONDRED	95,80	58,24	96,55	82,98
BETAGAMMA	96,36	58,24	96,55	83,67
MIFS	96,89	55,15	88,68	83,27
CMI	97,07	58,26	87,00	82,98
MIM	96,76	58,24	95,21	83,07
RELIEF	95,95	60,80	97,50	79,53
FCBF	96,15	55,64	78,28	83,27
MRF	95,56	58,20	78,28	78,77
SPEC	95,79	57,53	80,33	79,53
T-TEST	93,25	59,95	88,56	79,53
KRUSKAL-WALLIS	95,99	56,73	97,85	82,98
FISHER	96,76	59,95	96,82	79,53
GINI	96,76	58,83	96,95	83,07
GA	96,32	57,05	95,96	83,07

5.1 Comparison and Discussion

The figures 2,3,4,5 show the stability criteria for each feature selection techniques over the four datasets. The results show that MRMR, JMI, DISR, CIFE, ICAP, CONDRED, KRUSKAL-WALLIS and MRF have a stability value around 1.

This means that these feature selection criterions have selected the same feature selection for all the training sets in each data set. Figures 6,7,8,9 illustrate the stability criteria versus the number of features.

The analysis of the results indicates that the Relief, GA, and Ficher methods exhibit lower stability (measured by Kurcheva's, Information, SS, and SH metrics) across all datasets. Therefore, we conclude that these techniques are instable compared to the MRMR, JMI, DISR, CIFE, ICAP, CONDRED, KRUSKAL-WALLIS and MRF which have given an average stability close to 1.

The classification accuracy rate represents an important term to evaluate the performance of feature selection techniques.

In the figure 10 describes the classification accuracy rate for each number of features obtained by each feature selection criterions.

In term of classification accuracy rate, we show clearly that the both Spectrum and MRF methods have provided the lower classification accuracy rate. The higher accuracy rate for the WDBC data set is reached by the both JMI and MIM methods with 5 features.

For the ILPD dataset, we record the high accuracy for the CMIM and CIFE methods with 6 features. In the Cardiotocography data set, the high classification accuracy rate is achieved with Fisher score by using 13 features. For the Mammographic Mass data set, we record high accuracy for the CMIM and JMI methods with 2 features.

The figure 11 illustrates the classification accuracy rate obtained by the four better features selected by these methods in each test. We use the hold out method to generate 20 training sets for each data sets and we calculate the classification accuracy rate for each training sets by using the four better features.

There is different interpretation; each feature selection method is adapted to a special data set. We calculate the average classification accuracy rate obtained in each test and we summarize the results in the following table.

The goal of feature selection is to achieve a balance between the stability of a criterion and the classification accuracy rate (Gulgezen et al. 2009). This is why, experimental protocol was to take the average classification accuracy rate obtained by the 20 training sets plotted with the Kuncheva's Stability, Information Stability, SS Stability and SH Stability. Figures 12, 13, 14, 15 show the stability criterions versus the means accuracy rate. The goal is the find the set of feature selection criterions which the higher classification accuracy rate and the higher stability, this set is called the Pareto-Optimal Set.

The criteria which belonging the Pareto-Optimal set is said to be non-dominated [18]. Hence, it is evident from each subplot of Figures 12, 13, 14, and 15 that feature selection techniques positioned towards the top right of the space dominate over those towards the bottom left. Given this observation, there is no justification for selecting techniques located at the bottom left [18].

6 Conclusion

This paper introduces a comparison protocol evaluating twenty feature selection techniques across four datasets sourced from the UCI machine learning repository. The experimentation assesses stability criteria and classification accuracy rates calculated using SVM-SMO. Based on this research, we have concluded that each feature selection method can be tailored to suit specific datasets, considering factors such as the number of features and their distribution in the feature space.

The classification accuracy rate and the Stability provide a good experimentation and perfect information of features, the better feature selection method is one that has the both higher accuracy rate and stability. It is very interesting to evaluate the performance of these feature selection techniques in the analysing DNA Microarrays, where there are many features and comparatively few samples.

References

1. **Medjahed, S. A., Ouali, M., Benyettou, A., Ait-Saaid, T. (2015).** An optimization-based framework for feature selection and parameters determination of SVMs. International Journal of Information Technology and Computer Science, Vol. 7, No. 5, pp. 1–9. DOI: 10.5815/ijitcs.2015.05.01.
2. **Guyon, I., Elisseeff, A. (2003).** An introduction to variable and feature selection. International Journal of Machine Learning Research, Vol. 3, pp. 1157–1182.
3. **Goswami, S., Chakrabarti, A. (2014).** Feature Selection: A Practitioner View. International Journal of Information Technology and Computer Science, Vol. 6, No. 11, p. 66. DOI: 10.5815/ijitcs.2014.11.10.
4. **Lin, S. W., Lee, Z. J., Chen, S. C., Tseng, T. Y. (2008).** Parameter determination of support vector machine and feature selection using simulated annealing approach. Applied soft computing, Vol. 8, No. 4, pp. 1505–1512**.** DOI: 10.1016/j.asoc.2007.10.012.
5. **Kittler, J. (1978).** Feature selection and extraction, Academic Press, New York, https://api.semanticscholar.org/CorpusID:60554370.
6. **Rendell, L., Seshu, R. (1990).** Learning hard concepts through constructive induction: Framework and rationale. Computational Intelligence, Vol. 6, No. 4, pp. 247–270. DOI: 10.1111/j.1467-8640.1990.tb00298.x.
7. **Cortes, C., Vapnik, V. (1995).** Support-vector networks. Machine learning, Vol. 20, pp. 273–297. DOI: 10.1007/BF00994018.
8. **Bartlett, P., Shawe-Taylor, J. (1999).** Generalization performance of support vector machines and other pattern classifiers. Advances in Kernel methods—support vector learning, pp. 43–54.
9. **Keerthi, S. S., Shevade, S. K., Bhattacharyya, C., Murthy, K. R. K. (2001).** Improvements to Platt's SMO algorithm for SVM classifier design. Neural Computation, Vol. 13, No. 3, pp. 637–649. DOI: 10.1162/089976601300014493.

10. **Platt, J. C., Schölkopf, B., Burges, C., Smola, A. (1999).** Fast training of support vector machines using sequential minimal optimization. Advances in Kernel Methods - Support Vector Learning. DOI: 10.7551/mitpress/1130.003.0016.
11. **Flake, G. W., Lawrence, S. (2002).** Efficient SVM regression training with SMO. Machine learning, Vol. 46, pp. 271–290. DOI: 10.1023/A:1012474916001.
12. **Lewis, D. D. (1992).** Feature selection and feature extraction for text categorization. Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, pp. 23–26.
13. **Fleuret, F. (2004).** Fast binary feature selection with conditional mutual information. Journal of Machine Learning Research, Vol. 5, No. 9, pp. 1531–1555.
14. **Yang, H. H., Moody, J. (1999).** Data visualization and feature selection: New algorithms for non-gaussian data. Advances in Neural Information Processing Systems, pp. 687–693.
15. **Meyer, P., Bontempi, G. (2006).** On the use of variable complementarity for feature selection in cancer classification. Evolutionary Computation and Machine Learning in Bioinformatics, pp. 91–102. DOI: /10.1007/11732242_9.
16. **Lin, D., Tang, X. (2006).** Conditional infomax learning: An integrated framework for feature extraction and fusion. In: Leonardis, A., Bischof, H., Pinz, A. (eds). Computer Vision ECCV 2006, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg. Vol 3951. DOI: 10.1007/11744023_6.
17. **Jakulin, A. (2005).** Machine learning based on attribute interactions. PhD thesis, University of Ljubljana, Slovenia.
18. **Brown, G., Pocock, A., Zhao, M. J., Luján, M. (2012).** Conditional likelihood maximization: a unifying framework for information theoretic feature selection. The journal of machine learning research, Vol. 13, No. 1, pp. 27–66.
19. **Battiti, R. (1994).** Using mutual information for selecting features in supervised neural net learning. IEEE Transactions on neural networks, Vol. 5, No. 4, pp. 537–550. DOI: 10.1109/72.298224.
20. **Peng, H., Long, F., Ding, C. (2005).** Feature selection based on mutual information: Criteria of max dependency, max-relevance, and min-redundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 27, No. 8, pp. 1226–1238. DOI: 10.1109/TPAMI.2005.159.
21. **Yu, L., Liu, H. (2004).** Efficient feature selection via analysis of relevance and redundancy. The Journal of Machine Learning Research, Vol. 5, pp. 1205–1224.
22. **Zhao, Z., Liu, H. (2007).** Spectral feature selection for supervised and unsupervised learning. Proceedings of the 24th international conference on Machine learning, pp. 1151–1157. DOI: 10.1145/1273496.1273641.
23. **Wei, L. J. (1981).** Asymptotic conservativeness and efficiency of Kruskal-Wallis test for k dependent samples. Journal of the American Statistical Association, Vol. 76, No. 376, pp. 1006–1009. DOI: 10.1080/01621459.1981.10477756.
24. **Duda, R. O., Hart, P. E., Stork, D. G. (2001).** Patter Classification, Jhon Wiley and Sons, New York.
25. **Cover, T. M., Thomas, J. A. (1991).** Elements of Information Theory, Wiley.
26. **Cheng, Q., Zhou, H., Cheng, J. (2011).** The Fisher-Markov selector: fast selecting maximally separable feature subset for multiclass classification with applications to high-dimensional data. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 33, No. 6, pp. 1217–1233. DOI: 10.1109/TPAMI.2010.195.
27. **Zhao Z., Morstatter F., Sharma S., Alelyani S., Anaud A., Liu, H. (2010).** Advancing feature selection research. Feature Selection Repository Arizona State University.
28. **Oh, I. S., Lee, J. S., Moon, B. R. (2004).** Hybrid genetic algorithms for feature selection. IEEE Transactions on pattern analysis and machine intelligence, Vol. 26, No. 11, pp. 1424–1437. DOI: 10.1109/TPAMI.2004.105.
29. **Kalousis, A., Prados, J., Hilario, M. (2005).** Stability of feature selection algorithms. Fifth

IEEE International Conference on Data Mining ICDM'05, IEEE, pp. 8. DOI: 10.1109/ICDM.2005.135.

30. **Dunne, K., Cunningham, P., Azuaje, F. (2002).** Solutions to instability problems with sequential wrapper-based approaches to feature selection. Journal of Machine Learning Research, Vol. 1, pp. 22.
31. **Kuncheva L. I. (2007),** A stability index for feature selection. Proceedings of the IASTED International Multi-Conference on Artificial Intelligence and Applications, pp. 390–395.
32. **Yu, L., Liu, H. (2004).** Efficient feature selection via analysis of relevance and redundancy. The Journal of Machine Learning Research, Vol. 5, pp. 1205–1224.
33. **Yu, L., Ding, C., Loscalzo, S. (2008).** Stable feature selection via dense feature groups. Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 803–811. DOI: 10.1145/1401890.1401986.
34. **Fonseca, C. M., Fleming, P. J. (1996).** On the performance assessment and comparison of stochastic multiobjective optimizers. International conference on parallel problem solving from nature, Springer Berlin Heidelberg. pp. 584–593. DOI: 10.1007/3-540-61723-X_1022.

Article received on 20/12/2023; accepted on 16/04/2024.
**Corresponding author is Seyyid Ahmed Medjahed.*

ISSN 2007-9737

Proactive Load Balancing to Reduce Unnecessary Thread Migrations on Chip Multi-Processor (CMP) Systems

Ulises Revilla-Duarte[1], Marco A. Ramírez-Salinas[1,*], Luis A. Villa-Vargas[1], Andrei Tchernykh[2]

[1] Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico

[2] Centro de Investigación Científica y de Educación Superior de Ensenada, Departamento de Ciencias de la Computación, Mexico

{mars, lvilla}@cic.ipn.mx, chernykh@cicese.mx, urevillaa09@sagitario.cic.ipn.mx

Abstract. For a Linux operating system scheduler that is aware of Chip Multi-Processor (CMP) systems to carry out load balancing is extremely important and quite challenging. The scheduler is a vital component of the Linux kernel responsible for choosing the next thread to run and allocating to a processor core for execution. This process involves primarily a load-balancing procedure that provides the thread migration between the cores of a CMP system. A modern Linux scheduler is designed to obtain the best possible performance while ensuring a fair allocation of the processor cores' time among the normal (non-real-time) threads, which is known as Completely Fair Scheduling (CFS) policy. However, this policy collaterally can cause a relentless execution of the load-balancing procedure, and therefore, an excessive number of thread migrations. According to the literature, an increased cache invalidation, scheduling latency, and power consumption are issues inherent to this. In this paper, we propose and evaluate a proactive load-balancing (PLB) algorithm to reduce unnecessary thread migrations on CMP systems. By comprehensive experimental analysis, we show that our PLB algorithm reduces the number of thread migrations by 43.8% on average without degradation of performance.

Keywords. Linux CFS, load balancing, perf_event tool, PMU counters, chip multi-processor.

1 Introduction

The scheduler is a crucial component of the Linux kernel responsible for choosing the next thread to run and allocating to a processor core for execution [35, 37, 1, 13, 22]. This process involves primarily a load-balancing procedure that provides the thread migration between the cores of a CMP system. For a modern Linux scheduler that is aware of CMP systems to carry out load balancing is extremely important and quite challenging.

"The load-balancing procedure is based on a number of criteria of varying relative importance. The scheduling algorithm policy determines the importance of each of the criteria. Unfortunately, it is impossible to design an algorithm that fits in all the criteria simultaneously; trying to improve performance according to one criterion would adversely affect the expected performance by another" [14].

Nowadays, the Linux scheduling policy is designed to obtain the best possible performance while ensuring a fair allocation of the processor cores' time among the normal (non-real-time) tasks[1]. It is known as Completely Fair Scheduling (CFS) policy [20, 24, 28]. However, this policy

[1] Linux uses the term "task" to refer to both an entire process and a process thread.

ISSN 2007-9737

collaterally can cause a relentless execution of the load-balancing procedure and an excessive number of thread migrations [21].

It results in several disadvantages, such as an increased cache invalidation, scheduling latency, and power consumption [8, 21]. Let us briefly expose how the Linux scheduler's main functions work to gain a better understanding of the leading role of load balancing in CFS performance. A modern Linux kernel scheduler is composed of two functions: `scheduler_tick()` and `schedule()`.

The `scheduler_tick()` function is used by the kernel's timer system to periodically update the process' runtime statistics as well as to mark processes needing rescheduling (e.g., a higher priority task has just showed up, or a running task has simply spent too much time on a core). It is named the Periodic Scheduler.

The `schedule()` function is called by `scheduler_tick()` after a current process has been marked as needing rescheduling to fairly decide which process most deserves to run next. The current task itself may also call `schedule()` when it has to wait for a resource or an event's non-blocking signal in order to voluntarily yield, in the meantime, its core's time to another task.

A task temporarily yields its core's time without being blocked—the task remains in the TASK_RUNNING state—by calling the `sched_yield()` system call which ends up calling `schedule()` (ergo, Linux is a preemptive multitasking operating system).

The `schedule()` function is named the Main Scheduler. It is aware of CMP systems (a.k.a. homogeneous or symmetric multi-core systems). In the process of choosing the next task to run, `schedule()` carries out load balancing of both real-time (RT) and normal (CFS) tasks.

RT tasks are assigned the highest static priorities in the system (by default range from 0 to 99) in order to receive enough processing time to meet critical time constrains.

`schedule()` calls the function `pull_rt_task()`[2] to pull RT tasks from busier cores and distribute them according to their priorities among a group of RT subqueues (`struct rt_rq`) embedded as a field in the current core's run queue (`struct rq`).

CFS tasks are user tasks (including those of `root`) and kernel daemons that share a processor core according to their dynamic priorities given by nice values (numbers from -20 to 19 with a default of 0). As soon as `schedule()` is called, it disables the kernel preemption making sure not to be interrupted.

A run queue (`struct rq`) is a per-core, linear set of fields holding different types of data and statistics to handle the core's runnable tasks. The run queue is the primary scheduling data structure on which the Linux scheduler operates.

On the per-core run queue a CFS subqueue (`struct cfs_rq`) is built as a red-black tree data structure where tasks are arranged according to their runtime (given by the `vruntime` parameter). Tasks that have not run a relative long time are placed on the lower-left side of the tree. The left-most task is always picked to run next. Also, the run queue holds one RT subqueue (`struct rt_rq`) implemented as a doubly linked list per static priority level (0-99).

To balance CFS tasks, the Main Scheduler first checks the per-core `cfs_rq->nr_running` flag for load imbalance. This flag keeps track of the number of ready-to-run CFS tasks queued in a core's CFS subqueue. Then, `schedule()` calls the function `idle_balance()` which calls the function `load_balance()` to pull CFS tasks from bustling cores and insert them into the CFS subqueue (`struct cfs_rq`) in the current core's run queue (`struct this_rq`).

Once this pull-load balancing is done, the Main Scheduler picks the next task to run and performs the context switch. At this point, the Main Scheduler must be sure that there are no RT tasks in the current run queue waiting to be dispatched.

[2] From kernel versions 2.6.27 to 3.14.79 `pre_schedule_rt()` was used as an enveloping function for `pull_rt_task()`. Recently, from version 3.15.10 to the current stable version 4.16.6, `pull_rt_task()` is included in the `pick_next_task_rt()` function and invoked prior to pick the next rt task to run.

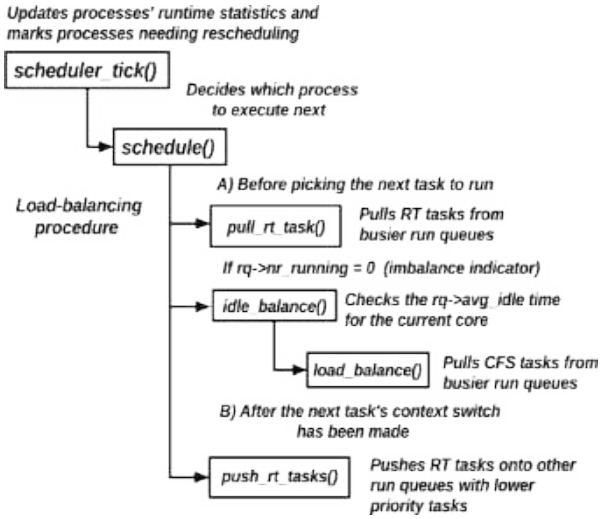


Fig. 1. Main scheduler's functions involved in load balancing

In order to be executed promptly, the Main Scheduler calls the function `push_rt_tasks()`[3] to push RT tasks, if any, from the current core's run queue onto the run queue of other cores with lower priority tasks (or even experiencing a lack of tasks).

Just after this push-load balancing is done, the Main Scheduler becomes preemtable and checks if the reschedule flag bit (`TIF_NEED_RESCHED`) in the thread information structure (`struct thread-info`) of the current task is set. If set, the search for a new task starts over. If not, `schedule()` exits.

Main Scheduler's functions involved in load balancing are shown in Fig.1. When a processor core is allocated to a new task, the previous task that was running on that core has either gone to the ready or blocked state [31], and can be migrated to another core when a new pull-load balancing operation is performed.

[3] From kernel versions 2.6.27 to 3.13.11 `post_schedule _rt()` was used as an enveloping function for `push_rt_tasks()`. The Main Scheduler now invokes `push_rt_tasks()` through the `balance_callback()` function just after context switching (from version 3.14.79 to the current stable version 4.16.6). `balance_callback()` is part of a novel mechanism which has added a new field for a `callback_head` data structure straight in the core's run queue. `struct callback_head` includes a `void (*func)` field that allows a faster handling of the callback functions `push_rt_tasks()` and `pull_rt_task()`.

Whenever load imbalance is detected, the load-balancing procedure is triggered to distribute the system load among the cores in a homogeneous multi-core processor, which results in excessive task migrations and the consequent drawbacks mentioned earlier.

On the other hand, in accordance with the literature [6, 11, 33, 44], threads' contention for shared resources on a multi-core processor is the major cause of system performance drop. This paper proposes and evaluates a proactive load-balancing (PLB) algorithm for Linux on CMP systems to avoid a decrease in performance due both to contention among the threads for shared resources and excessive thread migrations.

Our PLB algorithm keeps a high level of system performance by proactively averting contending threads from running concurrently as well as by reducing unnecessary thread migrations at runtime. We propose runtime-updated IPC thresholds, which are the basis of the operation of the proactive load balancing. Also, a complementary support algorithm to migrate threads on CMP systems is designed.

The PLB algorithm takes advantage of the Performance Monitoring Unit (PMU) accessible from each core of a modern multi-core processor, and the perf_event tool, a powerful profiling subsystem included in the Linux kernel since version 2.6.31. To meet our proactive load-balancing criterion, our algorithm carries out the following actions at runtime:

1. Configuration of the per-core PMU counters to read different performance-event samples simultaneously at constant time intervals; namely, those corresponding to the Instructions Retired, Unhalted Core Cycles and Thread Migrations events.

   Instructions Retired is the number of fully executed instructions and Unhalted Core Cycles is the number of cycles executed on the core (when the core was not in HALT state), i.e. it shows the total elapsed cycles. These events are sampled for each application thread as a part of the workloads launched separately.

2. Obtaining the Instructions Per Cycle (IPC) statistic which reflects the system performance by using the samples of the Instructions Retired and Unhalted Core Cycles events. In this way, IPC is obtained as follows:

   IPC = instructions retired / unhalted core cycles

   The initial value for the IPC-event (or performance) threshold is set to the first obtained IPC value.

3. Comparison of the subsequent IPC values with the current IPC-event threshold while the workload is running.
4. Reacting proactively to avoid performance ramp-down based on the result of this comparison.

These actions allow to proactively decide whether a running thread must be migrated to another core, and whether its current performance threshold value (previously-sampled IPC value) needs to be updated depending on the result of the comparison.

If a current thread's IPC count is below of its corresponding performance threshold value, the thread is migrated. If this count is higher, then the current performance threshold value is updated to this new IPC count (i.e., runtime-updated IPC thresholds are used).

In this way, only when a runtime-updated IPC threshold is not reached, our algorithm triggers the migration of contending threads in order to find couples of co-running threads that do not contend (or contend as little as possible) for shared resources on the cores, and therefore, leading back to a high level of system performance.

We thereby say that the algorithm obeys a criterion that proactively avoids contending threads from running concurrently.

Threads composing each workload are initially bound to a single core (e.g., core0) as the startup configuration. Our PLB algorithm is implemented at user level, which is sufficient for the accurate assessment [2].

In Section 8, we present a comparative table of different workloads when they run on Linux, first using the original (unmodified) Linux load-balancing procedure, and then merging our algorithm into the Linux kernel. This table shows that the number of thread migrations is significantly reduced (by 43.8% on average) without harming system performance when the PLB algorithm is used.

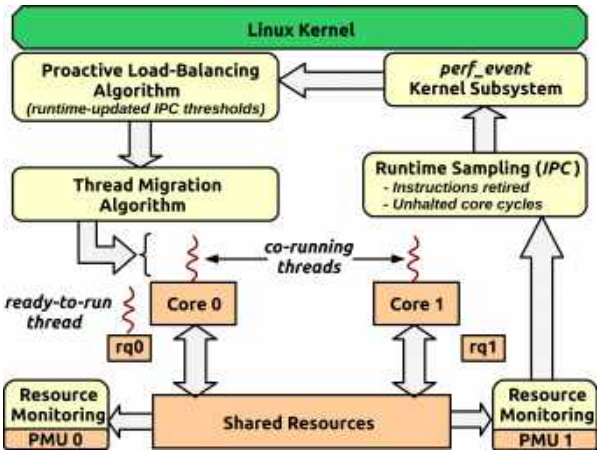


Fig. 2. Architecture of our proactive load balancer

Fig.2 illustrates the architecture of the proposed PLB algorithm. It shows its main components (rounded boxes) interacting with a dual-core CMP system (squared boxes). Next, we briefly describe each component addressed with more detail throughout this article.

- **Resource Monitoring through the PMU**: The PMU (Peformance Monitoring Unit) included within each core is composed of a special set of counting registers (Section 3) that we have configured to count the number of instructions that are fully executed (Instructions Retired), elapsed cycles (Unhalted Core Cycles) and (Thread Migrations).
- **Runtime Sampling (IPC)**: PMU registers are used to collect performance events sampled at regular time intervals at runtime (Section 3).
- **perf_event Kernel Subsystem**: PMU registers setup, runtime performance-event sampling and registers reading are all done through perf_event (Section 3).
- **Proactive Load-Balancing Algorithm**: Our algorithm uses runtime-updated IPC thresholds that indicate the minimum IPC values that must be reach at any sampling instant to hold the system performance at a steady high level.

Table 1. Main features of the example CMP system

Processor	Intel Core™i5 660 @ 3.33GHz
	Number of Cores 2 physical
	Number of Threads 4 (2 BIOS disabled)
	L1 Cache 2 x 32 KB
	L2 Cache 2 x 256 KB
	L3 Cache 4096 KB
Micro-architecture	Intel Westmere (Codename Clarkdale)
Main Shared Elements	L3 Cache, IM and PCI-e Controllers, and DMI and FDI interfaces
PMUs	Core, uncore and offcore MSRs register sets
Memory Size	4 GB
Memory Type	DIMM DDR3 Syncronous 1066 MHz (2 x 2GB)
Operating System	Ubuntu-GNU/Linux vanilla Linux kernel 3.1.2-SMP x86_32
Compiler	gcc version 4.5.2-8ubuntu4

Therefore, we rely on these threshold values to proactively decide whether to migrate a thread (Section 6).

– **Thread-Migration Algorithm**: If a thread needs to be migrated, our complementary support algorithm designed to migrate threads on a CMP system is invoked (Section 5). Next, the effect of this migration on the system performance is monitored and our PLB algorithm again decides whether to migrate a certain thread in order to maintain high system performance.

Table 1 summarizes the main features of the example CMP system used. The vanilla[4] Linux kernel version 3.1.2 is run on the Intel Corei5 660 processor with codename Clarkdale based on the Intel Westmere microarchitecture [7, 16] whose virtual cores (i.e., hyper-threading) have been disabled from the BIOS, thus having a CMP system with only 2 physical cores. The remainder of this paper is organized as follows: Section 2 briefly surveys related work. Section 3 describes at length the research framework used. Section 4 is devoted to workload selection.

[4]The standard Linux kernel available on the kernel.org web page.

Section 5 unveils the design stages of our thread migration support algorithm. Section 6 explains in detail the implementation of the algorithm that embodies our proposed proactive approach to perform load balancing. Section 7 delineates the evaluation experiments for the PLB algorithm. Section 8 reports the results. Finally, Section 9 concludes the paper and provides an avenue for future work.

2 Related Work

The design of scheduling algorithms that are aided by statistics collected via multi-core architecture-specific performance monitoring counters at run time to avert shared resource contention has been proposed in previous research. For this, they use either the Oprofile [26] or the Perfmon2 [29] external monitoring tools that are no part of the vanilla Linux kernel.

These algorithms aim to minimize the contention for the different shared resources within a CMP processor, such as the L2 or L3 caches, the system bus, the instruction queue, the core itself, and so on, therefore improving the overall system throughput.

Next, we present some previous work that has been done to implement scheduling algorithms that tackle the problem of shared resource contention in today homogeneous multi-core architectures:

Zhang X. et al. [43] developed a flexible framework for Throttling-Enabled Multi-Core Management (TEMM), which efficiently finds an optimal hardware throttling configuration for a user-specified resource management objective.

"It can support a variety of objectives for fairness, quality-of-service, overall performance, and power optimization. Throttling configuration refers to the settings of the platform-specific registers involved with the duty cycle modulation and dynamic voltage and frequency scaling (DVFS) mechanisms, originally designed for power management within processors.

TEMM searches for a reference configuration based on model predictions, and iteratively refines the search with a broad set of previously executed configuration samples. This search stops when a high-quality throttling configuration that meets the

objective is found". Sáez J.C. et al. [30] designed a non-work- conserving framework (i.e., a core may be idle at any time) to improve priority enforcement based on statistical information collected through hardware performance monitoring counters (PMU).

"When multiple threads run simultaneously, the system tries to detect changes in the behaviour of high priority (HP) threads that comes from negative interactions with other low-priority (LP) threads. Those changes trigger CPU disabling actions that temporarily block the potentially incompatible LP threads".

Herdrich A. et al. [15] adapted rate-based techniques (clock modulation and frequency scaling) that are employed to address power management and cache/memory Quality of Service (QoS) issues.

The QoS term refers to the ability to guarantee a certain level of performance. Basically, what they do is to regulate the time the core is active and/or its working voltage and frequency (DVFS technique) if it is running a low-priority task that harms the performance of a high-priority task due to system cache or memory contention.

Shi Q. et al. [32] proposed both a load-balancing algorithm based on the construction of scheduling domains by taking shared L2 cache into account and the design of load vectors to weigh the processor core's workload. Their goal is to reduce L2 cache misses (so main memory accesses are also reduced), and therefore, decrease the total execution time of threads.

Lim Q. et al. [21] implemented an operation-zone-based load balancer to improve the performance of multi-core systems at runtime. It provides three multi-core load-balancing policies based on the CPU employment.

"The cold zone policy loosely performs load-balancing operations; it is adequate when the CPU utilization of most tasks is low. The hot zone policy performs load-balancing operations very actively, and it is adequate for high CPU use. The warm zone policy takes the middle between the cold zone and the hot zone".

Our research work proposes a proactive approach to perform load balancing of software threads on homogeneous multi-core processors (i.e., CMP). Our proactive approach is primarily based on runtime-updated IPC thresholds that we devised and used in our decision-making model (Section 6) in order to reduce task migrations originated in the Linux scheduler.

On the example CMP machine used (Table 1), our PLB algorithm maintains two different threads from each workload running concurrently as long as it results in the least shared resource contention, and therefore, to the same extent, thread migration is reduced; thus helping to improve system performance.

Our work relies heavily on the performance monitoring subsystem of the Linux kernel, perf_event, to implement our routines that simultaneously monitor different performance events at runtime—thus providing valuable insight into how to use and configure perf_event.

In a first intance, we developed a complete workload-launcher tool that we used both to synchronously launch workloads made up of various CINT speccpu2000 benchmarks on the example multi-core system and to collect the resulting statistical data from a special set of performance-event counters located within each core's PMU (Performance Monitoring Unit) in the CMP processor.

Our results show that the number of migrations performed on the application threads (benchmarks) that make up the workloads used is significantly reduced (by 43.8% on average) without degradation of performance when our PLB algorithm is utilized.

3 Research Framework

This section describes the research framework used for the implementation of our PLB algorithm. Our research framework consists mainly of both the Performance Monitoring Unit (PMU) included in each core of a multi-core processor and the perf_event profiling tool available in the recent versions of the Linux kernel.

They are used jointly to implement our IPC-based decision-making model as well as to design the procedure for carrying out properly the sampling of different performance events simultaneously at runtime, which are the essential

Table 2. An excerpt from our code to collect instructions, cycles and cpu-migrations events in a single monitoring session

(1)	attr.type = PERF_TYPE_HARDWARE;
(2)	attr.size = sizeof(struct perf_event_attr);
(3)	attr.disable = 1;
(4)	attr.config = PERF_COUNT_HW_INSTRUCTIONS;
(5)	fd = perf_event_open(attr, pid, 0, -1, 0);
(6)	fd1 = perf_event_open(attr, pid, 1, -1, 0);
(7)	attr.config = PERF_COUNT_HW_CYCLES;
(8)	fd2 = perf_event_open(attr, pid, 0, fd, 0);
(9)	fd3 = perf_event_open(attr, pid, 1, fd1, 0);
(10)	attr.type = PERF_TYPE_SOFTWARE;
(11)	attr.config = PERF_COUNT_SW_CPU_MIGRATIONS;
(12)	attr.exclude_kernel = 0;
(13)	fd4 = perf_event_open(attr, pid, 0, fd, 0);
(14)	fd5 = perf_event_open(attr, pid, 0, fd1, 0);

parts of our algorithm. Programming details of the PMU counters and perf_event are also explained in this section. On the other hand, the different major program elements which make up the scheduler such as its main data structures and fuctions have been studied at length directly from the Linux kernel.

Basically, we mostly used the TOMOYO Linux Cross Reference [9], a very helpful web-based tool, to navegate and analyze extensively the vanilla kernel scheduler source code. Also, the Open MPI Portable Hardware Locality tool (`hwloc`) [5, 12] is first utilized to determine our system's topology and object numbering (`lstopo`), and then to bind threads onto processor cores (`hwloc-bin`).

3.1 The Performance Monitoring Unit

Processors supporting Intel 64 and IA-32 architectures have a Performance Monitoring Unit (PMU) consisting of a collection of Performance Monitoring Counter registers (PMCs) and Performance Monitoring Event registers (PMEs) [2, 25, 11, 17, 39]. PMCs and PMEs are implemented as Model Specific Registers (MSRs). They are accessed via the RDMSR and WRMSR instructions.

PMCs are used to collect event counts or serve as hardware buffers, so they are named Counter MSRs. PMEs are used to indicate what events need to be monitored, so they are named Event Programming MSRs. The number of MSR registers that make up the PMU depends on the processor model.

A monitor is defined to be a combination of a PME for the configuration and one or more PMC registers for collecting data. A counting monitor need only one PMC register.

Therefore, the counting monitors can each be programmed to count one event at a time. A monitoring session consists of several steps that must be followed to collect valid measurements. Those steps can be summarized as follows [25]:

i) Program the monitors (paired PME and PMC registers).

ii) Enable the monitors.

iii) Run the code to be monitored.

iv) Disable the monitors.

v) Collect results.

Each core built on a CMP chip has its own register bank which contains the MSR registers that make up the PMU [17]. The PMU and other registers in the register bank are grouped together to form the architectural state of a process thread.

That is, the architectural state is the set of registers within each core in the CMP processor that holds the state of its respective running subprocess. Therefore, on a CMP processor, each running thread has its own independent architectural state.

When a thread migrates from one core to another, its PMU state also moves. That is, counts of different events collected in the MSR registers in the source core's PMU are replicated into the same type of registers in the target core's PMU. It is this important design feature of multi-core processors that allowed us to implement code to follow a thread from one core to another without loosing information on the accounts of events.

Table 3. Workloads made up of CINT speccpu2000 benchmarks

CINT2000 workloads	
W1 (gzip gcc mcf)	**W11** (gzip bzip2 eon)
W2 (bzip2 gcc mcf)	**W12** (bzip2 eon crafty)
W3 (eon gcc mcf)	**W13** (gzip eon crafty)
W4 (crafty gcc mcf)	**W14** (bzip2 gzip crafty)
W5 (gzip gcc bzip2)	**W15** (eon gzip mcf)
W6 (eon gcc bzip2)	**W16** (bzip2 gzip mcf)
W7 (crafty gcc bzip2)	**W17** (crafty gzip mcf)
W8 (gzip gcc eon)	**W18** (eon bzip2 mcf)
W9 (crafty gcc eon)	**W19** (crafty bzip2 mcf)
W10 (gzip gcc crafty)	**W20** (crafty eon mcf)

3.2 The Linux perf_event Kernel Subsystem

Perf_event is a performance monitoring tool merged into the Linux kernel from version 2.6.31 [10, 40, 41]. The principal goal of perf_event is to provide Linux with the support needed to effectively utilize the PMU, thus allowing an advanced performance analysis.

Currently, it is a powerful kernel subsystem increasingly used in the research and development of new computer systems such as multi-core architectures. Support for the latest architectures is added according to new kernel versions.

The perf_event tool includes plenty of commands to collect and analyze performance and trace data. It can measure both hardware and sofware events. Software events are those that originate in the kernel. Some examples are: the number of context-switches, cpu-migrations or page-faults.

Hardware events are micro-architectural events such as the number of elapsed cycles, instructions retired, L1 cache misses, etc. The perf_event interface (API), `perf_event_ open()` (file `/tools/perf/perf.h`), wraps a single system call which supports a set of requests to configure, measure and collect performance monitoring information. It mainly provides a mechanism to read and write PMU registers.

By means of this system call, the PMU registers can be read during the execution of an application. Therefore, event samples can be obtained at runtime. This system call has the following prototype:

```
int perf_event_open(struct perf_event_attr
*attr, pid_t pid, int cpu, int group_fd,
unsigned long flags);
```

A description of its arguments can be found in the perf_event documentation (file `/tools/perf/Documentation`) and the references [10, 40]. The `perf_event_attr` structure is comprised of several attribute fields used to provide detailed configuration information for the event being created. The perf_event interface selects a PMU's counting monitor and configures its PME register based on the event to be monitored (given by the `attr.config` attribute).

It then returns an integer which is the file descriptor (`fd`) of the corresponding PMC register (counter) where the performance event counts will be collected. The `fd` is used to access the PMC register via standard system calls such as `read()` which is used to read the counter or `ioctl()` which is used to perform the counter input/output operations: reset, enable and disable.

Next, we show how the `perf_event_open()` system call is configured in order to implement a single monitoring session that collects hardware events such as `instructions` and `cycles` as well as a software event such as `cpu-migrations` simultaneously for both system cores.

An excerpt from our code to measure these events is shown in Table 2. Line (1) specifies the `type` attribute of the events to be measured which can be hardware or software type. As `instructions` and `cycles` are collected the hardware type is specified.

Line (2) sets the `size` attribute to the `attr` structure size. Line (3) sets the `disable` attribute to its default value, which is 1, to emphasize that the counter must start out disabled (due to synchronization reasons as discussed in Section 6). Then, line (4) introduces the `config` attribute which is nothing else but the name of the event to be measured. This attribute is set to collect the `instructions` event.

ISSN 2007-9737

Lines (5) and (6) define the system calls used to count the number of instructions retired on both cores (the third argument is `cpu = 0` for core0 and `cpu = 1` for core1). Additionally, the `group_fd` argument has been set to -1 to establish fd and fd1 as the group leaders for core0 and core1 respectively. Group leaders are used to collect different events as a unit for the same set of instructions that are fully executed (i.e., instructions retired). Line (7) shows the corresponding value for the `config` attribute to measure the `cycles` event. Lines (8) and (9) define the system calls used to count the number of elapsed cycles on both cores. Here, the `group_fd` argument is set to `fd` and `fd1`, the file descriptors for the group leaders.

Next, both the `type` and `config` attributes are changed to measure the `cpu-migrations` event. Line (10) now specifies the software type. Line (11) shows the right name for this event. As `cpu-migrations` is an event that happens in kernel space[5] (ergo, also recorded in the `se.nr_migrations` field of the task descriptor), the `exclude_kernel` attribute is changed from its default value of 1 to 0 to include events taking place in kernel space.

Line (12) shows the new value for this attribute. The system calls to measure `cpu-migrations` on both cores also have the `group_fd` argument set to `fd` and `fd1` as shown in lines (13) and (14). Thus, the `instructions`, `cycles` and `cpu-migrations` events are collected as a unit for the same set of instructions retired. Finally, for all events, the `pid` argument is set to the id number of the process thread to be monitored and the `flags` argument is set to zero.

4 Workload Selection

Workloads made up of different combinations of three CINT speccpu2000 benchmarks [36] were previously characterized using the vanilla Linux kernel 2.6.32.10 patched with the Perfmon2 profiling tool [29] on an Intel Core2 Duo E6550 multi-core processor [16].

[5]Linux divides virtual address space into two parts known as kernel space and user space (also called kernel mode and user mode respectively).

Table 3 presents our workloads and Fig.3 shows the bar graphs that result from their characterization using some key metrics. These metrics are:

a) **Instructions Per Cycle (IPC)**: fully executed instructions divided by the total CPU cycles: IPC = instructions retired / unhalted core cycles

b) **L2_MISSES**: data and instruction misses at second level (L2) cache. On the Intel Core2 Duo processor, the L2 cache is a unified cache that is shared by both cores to serve L1 cache misses of instructions and data.

c) **BUS_TRANS_MEM:BOTH_CORES**: Memory Bus Transactions due to both cores. That is, memory requests initiated by any core on the system bus.

d) **INST_QUEUE:FULL**: cycles during which the instruction queue is full. The instruction queue is a unit where instructions wait until they are ready for execution. An instruction is ready for execution when its operands have already been computed.

As can be seen in Fig.3, a smaller number of instructions per cycle is executed for workloads W1 to W4 and W15 to W20. Also, the number of L2 cache misses is too large for such workloads. Furthermore, both the number of memory requests and the number of cycles during which the instruction queue is full are also too large for these same workloads.

This indicates that the benchmarks composing workloads W1 to W4 and W15 to W20 contend with at least one of their co-runners for shared resources intensely. In particular: the L2 cache, the system bus and the instructions queue. Therefore, such workloads are regarded as best suited to carry out experiments in which a stress capacity for our CMP system is required.

5 Design of the Complementary Support Algorithm to Migrate Threads

The kernel uses the `sched_setaffinity()` system call to provide a different mask of cores (`new_mask`) to a task.

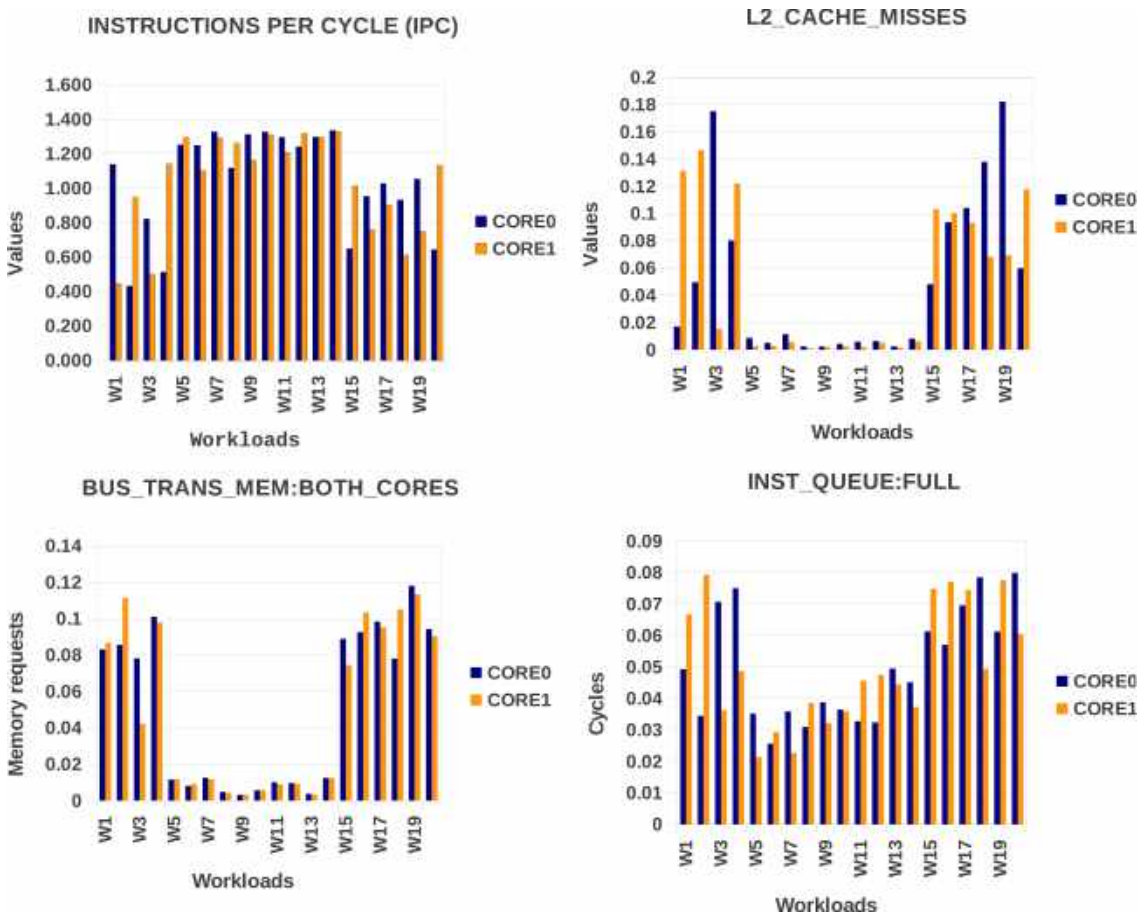


Fig. 3. Some metrics used for the CINT Speccpu2000 benchmarks characterization

First, this system call obtains a cpumask bitmap called `cpus_allowed` from the thread's task descriptor (`task_struct` structure) [6]. On the `cpus_allowed` bitmap one bit is set for each online core on which the thread can run (inherited from the process of which it is a part). Then, a bitwise AND operation is performed between the `cpus_allowed` and an input mask (`in_mask`) bitmaps to obtain `new_mask`[7].

If the core to which the current task is bound is not part of the new mask, `sched_setaffinity()` performs all the migration process[8]. The `sched_ setaffinity()` function receives a task id and an input mask as parameters. As a task id is assigned, it is necessary to obtain the corresponding task descriptor (since the `cpus_allowed` bitmask is there).

This is done through the `find_process_by_pid()` function called by `sched_ setaffinity()`. The various stages involved in the implementation of our algorithm that carries out the migration of threads between

[6] `sched_setaffinity()` calls the `cpuset_cpus_allowed()` function, which ends up calling the `task_cs()` function with a pointer to `task_descriptor` as parameter to retrieve the `cpuset` structure for the task. The `cpuset` structure holds the `cpus_allowed` cpumask bitmap. Since version 4.0.9, this structure includes the `effective_cpus` cpumask bitmap which is used instead of `cpus_allowed` for this very purpose.

[7] Through `cpumask_and()` which is called by `sched_ setaffinity()`.

[8] `sched_setaffinity()` calls the `set_cpus_allowed_ptr()` function to perform the entire process of migrating the thread when the core it is executing on is removed from the allowed bitmask.

Algorithm 1: mctopology algorithm

```
Input: Input mask in hex
Output: Length
Arguments: *buffer: user space data string
           length: length of the entered data string
1  Cpumask-type array: A-domain-core, B-domain-core;
2  procedure MCTOPOLOGY(mask);
3  Charater array: in_mask;
4  Integer variables: new_mask, num_cpus, cpui, A-domain;
5  Pointer to character: *mask_string;
6  num_cpus⟵get the number of cpus in the system (NR_CPU) ;   ▷ NR_CPU is a kernel variable
7  in_mask⟵get mask from user space (length of buffer);
8  mask_string⟵get address of (in_mask);
9  change what's in mask_string to hex;
10 save mask_string in new_mask;
11 for cpui⟵0, num_cpus do
12     if cpui is online then      ▷ its corresponding bit is set in the default kernel cpumask
13         right shift new_mask i positions;
14         do a bitwise AND operation between the;
15         right shifted new_mask and a 0×1 mask;
16         assign the result to A-domain;
17         if A-domain=1 then
18             save cpui in A-domain-core;
19         else
20             save cpui in B-domain-core;
21         Return length
```

the two physical cores of our example CMP processor (Table 1), which we have called `sched_setmigration_newmask()`, are described next. Likewise, our algorithm can be easily extended to a system with a larger number of cores. `sched_setmigration_newmask()` is invoked within our proactive load-balancing algorithm to migrate threads when needed. These stages are:

Stage 1: The Linux kernel's `sched_setaffinity()` function (file `/kernel/sched.c`) is modified so that it accepts a task descriptor as a parameter instead of the id of the corresponding task (hence, the bulky-code `find_process_by_pid()` function is removed). Since the `cpus_allowed` bitmask can be obtained from the task descriptor, this improves code complexity (ergo, power consumption also improves [4, 38]).

The `cpus_allowed` and input mask are used by the kernel to obtain the `new_mask` mask that the scheduler checks repeatedly to know which cores are offline and perform the thread migration process accordingly.

We have called the improved function `__sched_setaffinity()` (same system-call name prefixed with double underscore[9]), which is merged into our migration `sched_setmigration_newmask()` algorithm described in detail in Section 6. As we mentioned earlier in the Introduction, our work is based on the vanilla Linux kernel version 3.1.2. From version 5.15.67, the core part of the code within `sched_setaffinity()`[10] (now found in

[9] So it is treated as an internal function and no checks are made on the user address space (the function is then executed faster) [42].

[10] The essential code for obtaining the `cpus_allowed` cpumask bitmap and migrating.

Algorithm 2: sched_setmigration_newmask() algorithm

```
   Input: current task's pid, core number
   Output: retval;                                   ▷ 0 if success, error code if failure
 1 Cpumask-type array: A-domain-core, B-domain-core;
 2 procedure sched_setmigration_newmaskpid, core;
 3 Integer variable: retval;
 4 Pointer to current task: *curr;
 5 curr ⟵ get current task(pid);                     ▷ kernel's own function
 6 if core=0 then
 7 |   retval⟵call_sched_setaffinity(curr, A-domain-core);   ▷ core1 ∉ A-domain-core
 8 else
 9 |   if core=1 then
10 |   |   retval ⟵ call_sched_setaffinity(curr, B-domain-core);   ▷ core0 ∉ B-domain-core
11 |   else
12 |   |   Return error code
13 Return retval
```

/kernel/sched/core.c) is wrapped in a function that also bears the same name prefixed with double underscore (__sched_setaffinity()). However, sched_setaffinity() still uses the find_process_by_pid() function to obtain the thread's task descriptor.

Stage 2: Algorithm 1 shows our algorithm called mctopology (i.e., multi-core topology) which is designed to construct from an input mask in hexadecimal notation two domain masks of cores:

A-domain-core and B-domain-core

The input mask is devised to allow grouping cores that are physically adjacent (and so sharing resources, e.g., a common cache) into domain masks (i.e., scheduling domains) such that threads should preferably be moved between the cores in a domain. To pass the input mask to the kernel to be processed, the /proc/topology directory and the user_cpumask_input virtual file are created as a means of communication between the user space and the kernel space.

In lines 10 and 11, the number of cores in the system (taken straight from the kernel variable NR_CPU) is stored in the num_cpus variable and the input mask is stored in the in_mask variable respectively. In lines 12 – 14, in_mask's address is assigned to the mask_string pointer and the string value stored at the address where mask_string is pointing is changed into a hexadecimal format and stored in the new_mask variable. Lines 15 – 28 comprise the method for grouping cores into scheduling domains so that a thread is migrated to a core within the same domain. In this way, the thread's information stored in the cache that is shared among the cores within the domain is not moved to another system cache (resulting in lower migration cost).

This section of code first tests if the bit that corresponds to the core at position i (cpui) is set in the default kernel cpu_allowed cpumask (i.e., if the core is online). If so, a right shift operation on new_mask of i positions is performed and a bitwise AND operation between the right-shifted new_mask and a 0x1 mask (to select adjacent cores) is executed. Then, the result is assigned to the A-domain variable (similarly, a 0x2 mask could have been used for the B-domain variable).

Depending on the result that has been stored in A-domain, the core is saved in either the mask A-domain-core or B-domain-core. Since our system has only two (online) cores, the domain masks A-domain-core and B-domain-core store corei (i=0) and corei (i=1) respectively. Depending on the number of cores in a multi-core system, more complex layouts of scheduling domains can be obtained.

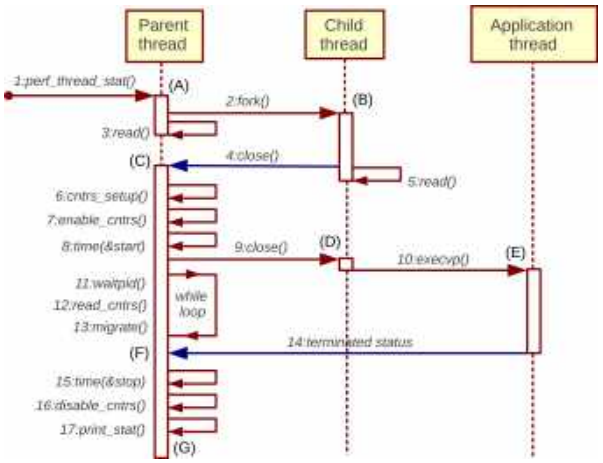


Fig. 4. Sequential diagram for our PLB algorithm

As the number of cores increases, the setting of an input mask so that the selection of the adjacent cores be optimal is a matter of key importance.

Stage 3: An algorithm to update the current task's cpumask bitmap with either the mask `A-domain-core` or `B-domain-core` is devised and implemented as a system call. It is invoked within our Proactive Load-Balancing algorithm to migrate threads when needed. We named it:

`sched_setmigration_newmask()`

Our `sched_setmigration_newmask()` algorithm receives, as a parameter, the number of the core to which the task must be migrated. Depending on the core number, the algorithm chooses either the `A-domain-core` or `B-domain-core` domain mask. The algorithm then calls the `__sched_setaffinity()` function with the current task descriptor and the new mask as parameters, so the current task is migrated to the target core.

For our example dual-core system (which has two physical cores and hyperthreading disabled), the default kernel cpumask is 0x11 (both cores are online), so when the `A-domain-core` domain mask is chosen within our `sched_setmigration_newmask()` algorithm, the current task is forced to leave core0 and migrate to core1. Similarly, when the `B-domain-core` domain mask is chosen, the current task is forced to leave core1 and migrate to core0. This algorithm is shown in Algorithm 2.

In line 7, the current task descriptor is obtained first. In lines 8 – 17, a domain mask for the `__sched_setaffinity()` function is chosen as its second parameter according to the core number that is passed to `sched_setmigration_newmask()`. Since our system has only two (online) cores, there is one core per domain.

On Linux, threads are always migrated from one run queue to another. Before being migrated, the kernel must suspend the execution of the thread running on the local core and save its task context (including performance statistics) accordingly. These actions are part of a context switching procedure performed in kernel mode. For a thread, its task context is obtained from its task descriptor (`task_struct` structure) and CPU registers[11].

The task descriptor contains all of the data needed to keep track of the thread, whether it is running or not. Some of the primary fields it includes are[12]: A `thread_info` structure that holds all requiered processor-specific low-level information about the thread, a `__state` variable, a `*stack` pointer to its kernel-mode stack, a `*mm` pointer to its virtual address space (`mm_struct` structure, also called memory descriptor), and a `thread_struct` structure that holds the architecture-specific state of the thread.

The thread's virtual address space[13] is divided into several regions of type `vm_area_struct` each of which contains different information of the running thread such as its user-mode stack, code, data, and so on. When entering kernel mode to run the context switcher, the instruction pointer (EIP), the status register (EFLAGS, also known as a condition-code register or CCR), the user stack pointer (EBP), the segment selector of the user data segment (__USER_DS) and the segment selector of the user code segment (__USER_CS) of the thread being migrated are saved automatically

[11] It is not necessary to save the full state of the machine for a thread, as it is using the same memory, program code, files and devices as the process of which it is a part. A thread must maintain only some state information of its own.

[12] From the latest stable kernel version 5.19.5.

[13] The process that spawns the thread initially shares its address space with it. Later, when the thread modifies or writes to a part of this space, a copy of that part is made for the thread itself (known as the Copy-On-Write technique) [34].

Algorithm 3: proactive load-balancing algorithm Part 1

Input: application's executable code and its own parameters
Output: performance events' statistics of the monitored application thread

```
1  procedure perf_thread_stat() ;                    ▷ input is entered at the command line
2  child_pid⟵fork()
3  if child_pid < 0 then                                          ▷ the parent does this
4      failed to fork
5  if child_pid ≠ 0 then                                           ▷ the child does this
6      do ⟶ preparatory steps prior to launching the application thread;
7      close one end of the pipe 1;
8      read the open end of pipe 2;
9      execvp ⟵ application's name;
10 read the open end of pipe 1;                                   ▷ the parent does this
11 do ⟶ the counters' settings;                                          ▷ See Table 2
12 ioctl⟵ RESET, fd_i;                          ▷ a file descriptor (fd) selects a counter
13 ioctl⟵ ENABLE, fd_i;
14 t_0 ⟵ time(& start);                                            ▷ the start runtime
15 close one end of pipe 2;
16 k ⟵ 0;
17 waitpid ⟵ child_pid, WNOHANG ; ▷ in order to not suspend the execution of the parent thread
```

on the kernel-mode stack[14]. The context switcher then saves in the `thread_struct` structure housed in the task descriptor, the remainder of CPU registers that hold the state of the machine at the time the core is deallocated from the thread.

Thus, since the thread's task descriptor is moved from the local run queue to the target run queue, the thread's task context can be restored and its execution resumed (a thread is always scheduled from the run queue on to the core in user mode [18]).

CPU cycles consumed in all this moving that the kernel does to migrate threads result in pure overhead, because no useful work (IPC) is done [34, 8, 22, 23].

In this light, our PLB algorithm maintains two different threads from each workload running concurrently as long as it results in the least shared resource contention, and therefore, to the same extent, thread migration is reduced; ultimately preventing system performance from declining.

[14] A context switch is always initiated by an interrupt. The interrupt mechanism saves automatically this data on the kernel-mode stack [18, 3, 42, 31].

In the next section, we first illustrate sequentially how our algorithm works, and then explain its pseudocode line by line.

6 Proactive Load-Balancing (PLB) Algorithm

6.1 Sequential Diagram

A sequential diagram to depict the interaction of the various threads involved in our algorithm is shown in Fig. 4. In this diagram the leading function (parent thread) called `perf_thread_stat()` appears in the upper left and tagged with the number 1. When the algorithm starts, it calls the function `perf_thread_stat()` which spawns a child thread using the kernel's standard `fork()` function.

Next, the child thread starts executing in parallel to the parent until the parent thread stops and waits for the child to complete the preparatory steps prior to launching the application (which will be detailed in the next section) before carrying out the counters' setup. Using an inter-thread communication technique known as pipes, the parent and child get synchronized by sending

Algorithm 4: proactive load-balancing algorithm Part 2

```
1  while do
2      if k = 0 then                                   ▷ first pass inside the while loop
3          refval1⟵ ipcval1;
4          refval2⟵ ipcval2
5      if k ≠ 0 then
6          if refval1 < ipcval1 then
7              refval1 ⟵ ipcval1
8          if refval2 < ipcval2 then
9              refval2 ⟵ ipcval2
10     if count2 ≠ 0 && ipcval1 < refval1 && mf1 ≠ 1 then
11         res ⟵ sched_setmigration_newmask(child_pid, 1);  ▷ our algorithm to migrate threads
12         mf1 ⟵ 1;
13         mf2 ⟵ 0;
14         refval1 ⟵ 0;                                    ▷ refval1 is reset
15     if count3 ≠ 0 && ipcval2 < refval2 && mf2 ≠ 1 then
16         res⟵ sched_setmigration_newmask(child_pid, 0);  ▷ our algorithm to migrate threads
17         mf2⟵ 1;
18         mf1⟵ 0;
19         refval2⟵ 0 ;                                    ▷ refval2 is reset
20     k ⟵ k + 1
21 t_1 ⟵ time(&stop);                                      ▷ the end runtime
22 ioctl⟵ DISABLE, fd_i;
23 read the final values of the statistics;
24 print the performance events' statistics of the monitored application thread;
```

messages between them. A null message is sent from the end of a pipe that gets closed, to the end that remains open to read the message. The `close()` function is used to close one end of a pipe, whereas the `read()` function is used to read at the other end.

This is shown through the points A-C. When the parent thread receives the null message from the child, the parent resumes execution and performs the counters' setup. Straight afterwards, the parent thread first resets and enables the counters and then starts measuring the runtime parameter through the `time()` function.

Meanwhile, the child thread waits for the parent to finish these steps. Right after the parent thread has started measuring the runtime parameter, it sends a null message to the child by closing one end of a second pipe (D).

The child thread reads the message at the other end of this pipe, resumes execution, and launches the application by means of the `execvp()` function. Now, this is shown through the points C-E. While the application is running, the parent thread reads the counters and stores their values into data arrays within a `while` loop.

The `while` condition includes a `waitpid()` function set to wait for the application to terminate without suspending the parent thread execution (by using the `WNOHANG` option). Thus, the parent thread keeps gathering statistical information from the counters while the application is running.

Based on these statistics, the parent thread decides whether it is convenient to migrate the current application thread. When the application ends, the terminated status for the child specified by its pid is immediatly available.

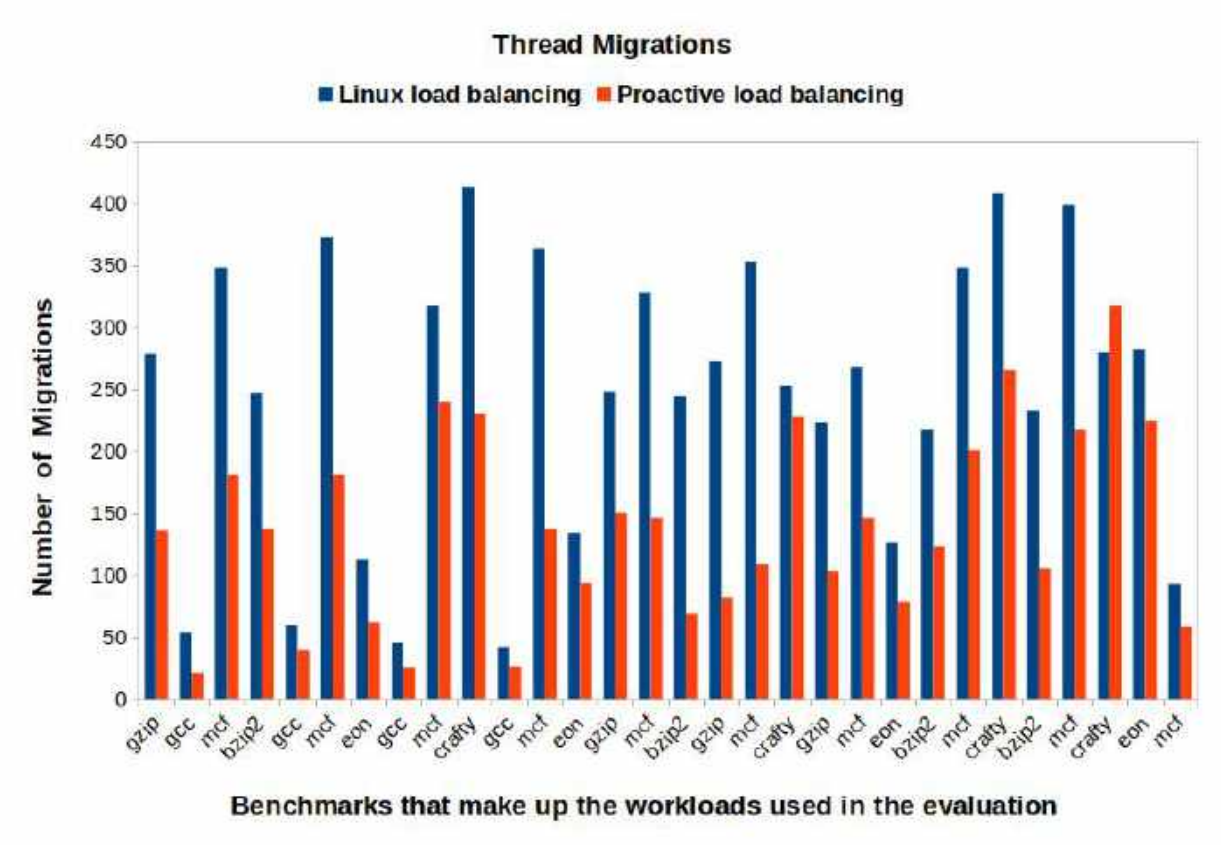


Fig. 5. Number of thread migrations obtained for both load balancers after 10 runs of each workload

Then, the parent thread leaves the `while` loop, stops measuring the runtime, and disables the counters. The `ioctl()` function was used to reset, enable, and disable the counters.

Finally, the parent thread saves the collected statistics in dedicated files, prints them on the screen, and exits. See points E-G. Next, we present our proactive load-balancing algorithm split into two parts: Algorithm 3 and Algorithm 4.

6.2 Detailed Description

Our PLB algorithm requires a tight synchronization between the parent, child and application threads as exposed so far. Algorithm 3 details the synchronization between these threads as well as the first part of the `while` loop depicted above.

In lines 4 – 8, the parent thread spawns a child thread through the `fork()` function and checks if the returned `pid` has a valid value.

At this point a branch occurs: in line 9, the child begins executing the preparatory steps prior to launching the application thread, and in line 14, the parent thread waits for the child to finish.

In line 10, just after the child thread has completed these steps, it sends a null message to the parent by closing one end of pipe 1. In line 11, the child waits for the parent thread to carry out the counters' setup.

Then, in lines 15 – 18, the counters are setup, reset and enabled by the parent thread, whereupon it begins measuring the runtime parameter. In line 19, the parent sends a null message to the child by closing one end of pipe 2, so in line 12, the child launches the application by means of the `execvp()` function.

Table 4. Percentage of reduction in the number of thread migrations for each workload

Thread Migrations									
Workload	Benchmarks (threads)								
W1	**gzip**			**gcc**			**mcf**		
	278	136	**51.1%**	54	21	**61.1%**	348	181	**47.8%**
W2	**bzip2**			**gcc**			**mcf**		
	247	137	**44.5%**	59	39	**35.4%**	372	181	**51.4%**
W3	**eon**			**gcc**			**mcf**		
	112	62	**44.6%**	45	25	**44.4%**	317	240	**24.3%**
W4	**crafty**			**gcc**			**mcf**		
	413	230	**51.6%**	42	26	**38.1%**	363	137	**62.2%**
W15	**eon**			**gzip**			**mcf**		
	134	94	**29.9%**	248	150	**39.5%**	328	146	**55.5%**
W16	**bzip2**			**gzip**			**mcf**		
	244	69	**71.7%**	272	82	**69.8%**	352	109	**69.0%**
W17	**crafty**			**gzip**			**mcf**		
	252	228	**9.5%**	223	103	**53.8%**	268	146	**45.1%**
W18	**eon**			**bzip2**			**mcf**		
	126	78	**38.1%**	217	123	**43.3%**	348	201	**42.2%**
W19	**crafty**			**bzip2**			**mcf**		
	408	265	**35.0%**	232	105	**54.7%**	398	217	**45.5%**
W20	**crafty**			**eon**			**mcf**		
	279	317	**-13.6%**	282	224	**20.6%**	93	58	**37.6%**

As mentioned earlier, the child performs a couple of preparatory steps prior to launching the application thread which consist in:

1. First calling a dummy `execvp()` that always fails in order to avoid Global Offset Table (GOT) and Procedure Linking Table (PLT) entry relocation overhead on the real `execvp()` [19, 27].

   These processor-specific tables assist the dynamic linker in finding the absolute addresses for position-independent function calls, such as `execvp()`. Therefore, as all this action is performed in advance, to launch the real `execvp()` takes much fewer steps.

2. Setting the close-on-exec flag (`FD_CLOEXEC`) associated with the file descriptor representing the open end of pipe 2, so this end will be automatically (and atomically) closed when the `execvp()` succeeds (since `execvp()` does not return when successful).

On the other hand, in line 20, the control variable `k` that is used to indicate the number of times the algorithm enters into the `while` loop is initialized to 0. In line 21, the `waitpid()` function is configured using the `WNOHANG` option along with the child's pid in order not to suspend the execution of the parent thread while the application is running. In lines 22 – 26, the `while` condition is set and the counts of the events represented by their corresponding file descriptors (Table 2) are stored into dedicated array variables. In lines 27 – 38, the IPC is calculated from the instructions retired and unhalted core cycles statistics for both the core0 and core1 at runtime.

The decimal part of the IPC values thus obtained is truncated to its hundredth part; this in order to make them more meaningful. Algorithm 4 presents the second part of the `while` loop which contains the reasoning for deciding to migrate the application threads (i.e., the decision-making part).

In lines 39 – 49, for the first pass inside the `while` loop, the IPC values that have been previously calculated are assigned to IPC-threshold variables (refval1 and refval2). If it is a subsequent pass, it checks whether the IPC-threshold values are lower than the corresponding new IPC values for core0 and core1.

If so, the new IPC values are assigned to the IPC-threshold variables (i.e., runtime-updated IPC thresholds are used). On the other hand, in lines 51 – 62, if the new IPC values are lower than the IPC-threshold values, it means that a significant contention exists between the co-runner threads.

So the current IPC-threshold values are maintained. In line 51, the IPC value of the thread running on core0 (`ipcval1`) is checked, if it is lower than its corresponding IPC-threshold value (`refval1`), the thread is migrated to core1 by means of our `sched_setmigration_newmask()` algorithm.

Similarly, in line 57, the IPC value of the thread running on core1 (`ipcval2`) is checked, if it is lower than its corresponding IPC-threshold value (`refval2`), the thread is migrated to core0[15]. Therefore, as soon as a significant decrease in the thread's IPC is observed at runtime, our algorithm reacts proactively trying to keep this parameter to its previous higher value by migrating the thread and thus avoiding contention for shared resources with its co-runners.

[15]This is completely in line with our proposed thread migration model. For a system with a larger number of cores, our model would simply have more choices of cores to migrate to.

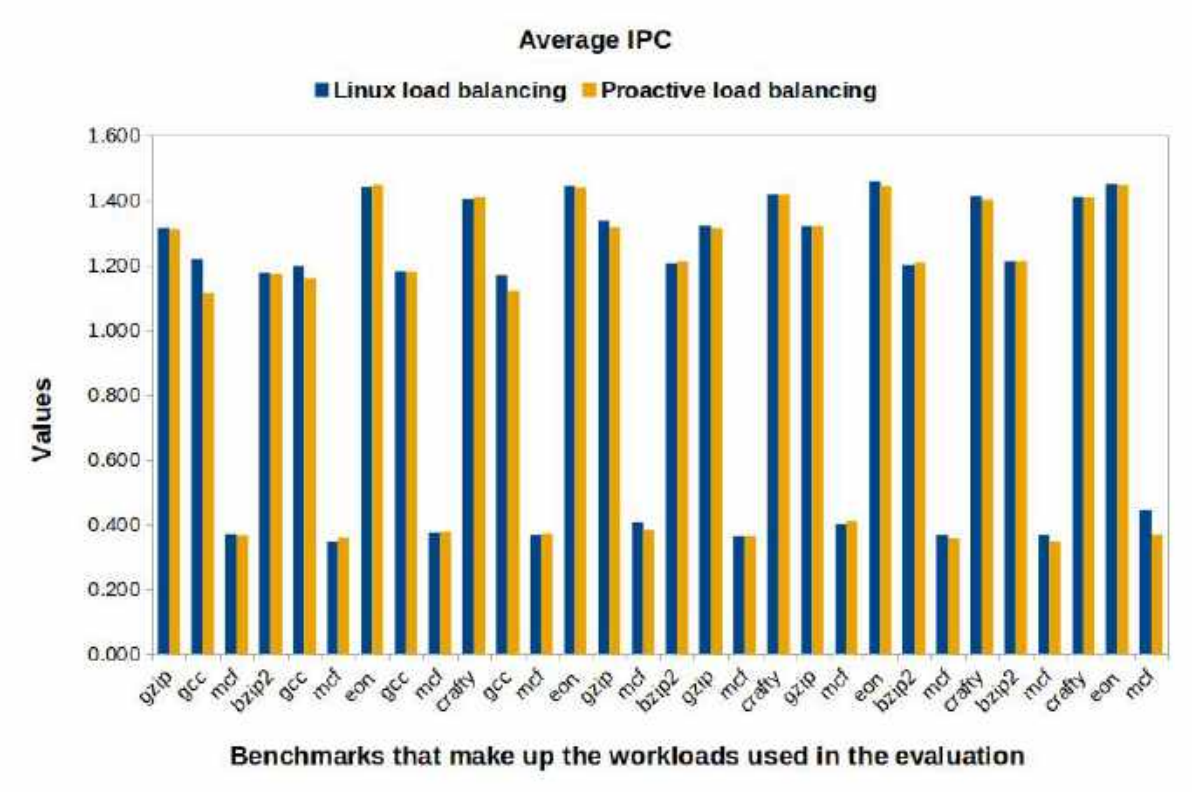


Fig. 6. Average IPC obtained for both load balancers after 10 runs of each workload

As previously stated, the optimal couples of co-runner threads are searched in this way at runtime. The `mf1` and `mf2` variables are used to ensure that a thread cannot be migrated to the same core where it is currently running. In line 63, the control variable `k` is incremented by 1 for each pass through the `while` loop.

Finally, when the application terminates, the `while` condition is no longer met, so in lines 65 – 68, the parent thread stops measuring the runtime parameter, disables the counters, reads the final values of the statistics, and prints these final values along with those previously obtained during the execution of the application.

7 Evaluation Experiments

This section details the steps performed to evaluate our proactive load-balancing algorithm: First, workload implementations of CINT speccpu 2000 benchmarks [36] which simulate application threads are carried out. Three-benchmark workloads that have adequate stress capacity for the example multi-core system used (Table 1) are employed. Second, perf_event is used to configure the PMU built into each processor core in order to collect the instructions retired and unhalted core cycles hardware events from which we calculate the Instructions Per Cycle (IPC) statistic.

As explained in earlier sections, IPC is the primary metric we employed to manage the migration of process threads. Although there are several other events collectable such as L2 cache misses, rejected L2 cache requests (by the bus queue), completed memory transactions on the system bus, and so on, IPC is a good performance metric to keep simplicity in our conception. The PMU is also configured to collect the cpu-migrations software event at the same time (as described in Section 3).

Third, as mentioned in Section 2, we developed a workload-launcher tool in order to synchronously launch the implemented workloads on the CMP system. The workload-launcher picks the different bechmarks that make up a workload and launches them simultaneously to execution. As soon as a workload finishes, the benchmarks that compose the next workload are launched. When the last workload is executed, the workload-launcher starts over by executing each workload again in a second round. It stops after 10 rounds are completed (i.e., each workload is run 10 times). As discussed earlier, 10 workloads were selected from the 20 that we had previously characterized to stress the CMP system used.

Fourth, in the manner described in the previous step, each of the selected workloads is run 10 times on Linux without modifying the scheduler, and the number of instructions retired, unhalted core cycles and thread migrations are counted at runtime. Fifth, our PLB algorithm is then merged into the Linux scheduler and each of these workloads is run 10 times again. The same events are counted at runtime. Finally, the statistics values obtained are plotted on a bar graph and analyzed. This bar graph is shown in Fig.5.

8 Results

The number of thread migrations obtained after running each workload 10 times on the example CMP system with the Linux OS installed, first with the vanilla (standar) kernel and then with the PLB algorithm merged into the Linux kernel scheduler is plotted in the bar graph in Fig. 5, which also shows the benchmarks that make up each workload. From this figure it can be seen that the number of migrations performed is significantly reduced when using proactive load balancing.

Table 4 shows in detail the percentage by which the number of migrations decreased for each constituent benchmark (mimicking a software thread) of the workloads used. In this table, below each benchmark, there are three small boxes. In the first box, we have the resulting number of thread migrations for the vanilla Linux scheduler, in the second, the corresponding number for our proactive load-balancing algorithm, and in the third, the percentage by which the migrations decreased. Our PLB algorithm reduces the number of thread migrations by up to 71.7% (`bzip2` in W16). There is only one case (`crafty` in W20) where there is a 13.6% increase. For this particular combination of benchmarks that make up W20, both `eon` and `mcf` contend, one at a time, against `crafty` for shared resources with great intensity.

Therefore, our algorithm is forced to migrate `crafty` quite often. For these workloads, we also compared the average IPC obtained for both the vanilla and modified scheduler instances. The resulting numbers for proactive load balancing are practically the same as those of the vanilla instance. This is exposed in the bar graph in Fig.6. From Table 4, we obtain the total number of migrations for both the vanilla scheduler (M_T) and proactive load balancing (m_T): Vanilla scheduler:

$$M_T = \sum_{i=1}^{30} M_i = 7354. \tag{1}$$

Proactive load balancing:

$$m_T = \sum_{i=1}^{30} m_i = 4130. \tag{2}$$

Thus, the total reduction in the number of migrations (T_r) is:

$$T_r = M_T - m_T = 7354 - 4130 = 3224. \tag{3}$$

That is, in total there were 3224 fewer migrations.

Therefore, on average the percentage by which the number of migrations was reduced (A_r) is:

$$A_r = \left(\frac{3224}{7354} \times 100\right)\% = 43.8\%. \tag{4}$$

Without degradation of performance.

9 Conclusion and Future Work

We have presented our proactive load-balancing (PLB) algorithm designed to avoid performance drop in CMP systems due both to an excessive number of thread migrations and shared resource contention. Our PLB algorithm proactively decides whether a running thread must be migrated from its current core to another active core based on performance data read at runtime from system counters (PMU) that we configured to collect counts of selected events.

While the software threads are running concurrently (co-running threads) on a CMP processor, the Instructions Retired and Elapsed Cycles events are collected and used to obtain the IPC parameter which shows the system performance. In the same way, the cpu-migrations event is also gathered to know the number of migrations performed on each thread.

According to the literature, shared resource contention by co-running threads is the most important cause of performance drop. Hence, our PLB algorithm is primarily designed to proactively avoid resource contention. On the example CMP machine, our algorithm maintains two different threads from each workload running concurrently as long as it results in the least shared resource contention, and therefore, to the same extent, thread migration is reduced.

Our PLB algorithm avails itself of the Linux kernel's perf_event subsystem to configure each core's PMU, read event counts and monitor software threads. The perf_event subsystem represents an easy access to hardware counters to Linux, which are a key resource for improving system performance. Unfortunately, there is a lack of literature and limited online documentation available for this monitoring tool.

Therefore, an important aspect of our research work is that we have excelled at dredging up most of the vague and obscure configuration facts, and thus shedding light on how to use and set up perf-event; namely, how to set it up in order to count different events simultaneously at runtime. We have introduced a comprehensive view of the methodology used for conducting research to adapt an operating system such as Linux to the modern multi-core architectures, which is a theme of great relevance and interest among computer scientists today. Overall, our results have shown that the number of migrations performed on the threads (benchmarks) that make up the workloads used is significantly reduced (by 43.8% on average) without harming system performance when our proposed PLB algorithm is utilized. To design A.I. algorithms that can be merged into Linux to perform smart scheduling of co-running threads in multi-core architectures so that system performance improves is an interesting avenue for future work.

Acknowledgments

We thank Instituto Politécnico Nacional (IPN) for funding this work through the projects:

– SIP20231335: Open Technologies for the Development of SoCs based on the Lagarto-I RISC-V Processor and Linux OS for Academia and Research.

– SIP20231075: Lagarto SBC: Computer on module based on the Lagarto processor.

Ulises Revilla-Duarte thanks Consejo Nacional de Ciencia y Tecnología de México (CONACYT) for the Ph.D. Scholarship granted and IPN for the facilities provided to carry out this research work.

References

1. **Arpaci-Dusseau, R. H., Arpaci-Dusseau, A. C. (2018).** Operating systems: Three easy pieces. Chapter 10: Multiprocessor Scheduling, CreateSpace Independ-ent Publishing Platform, pp. 103–112.
2. **Blagodurov, S., Fedorova, A. (2011).** User-level scheduling on NUMA multicore systems under linux. Proceedings of the Linux Symposium, pp. 81–92.
3. **Bovet, D. P., Cesati, M. (2005).** Understanding the Linux kernel. O'Reilly.

4. **Brandolese, C., Fornaciari, W., Salice, F., Sciuto, D. (2002).** The impact of source code transformations on software power and energy consumption. Journal of Circuits, Systems and Computers, Vol. 11, No. 5, pp. 477–502. DOI: 10.1142/s0218126602000586.
5. **Broquedis, F., Clet-Ortega, J., Moreaud, S., Furmento, N., Goglin, B., Mercier, G., Thibault, S., Namyst, R. (2010).** hwloc: A generic framework for managing hardware affinities in HPC applications. Proceedings of the 18th Euromicro International Conference on Paralel, Distributed and Network-based Processing, pp. 180–186. DOI: 10.1109/pdp.2010.67.
6. **Chandra, D., Guo, F., Kim, S., Solhin, Y. (2005).** Predicting inter-thread cache contention on a chip multi-processor architecture. Proceedings of the 11th International Symposium on High-Performance Computer Architecture, pp. 340–351. DOI: 10.1109/HPCA.2005.27.
7. **Clarkdale (2007).** Clarkdale microprocessor.
8. **Constantinou, T., Sazeides, Y., Michaud, P., Fetis, D., Seznec, A. (2005).** Performance implications of single thread migration on a chip multi-core. ACM SIGARCH Computer Architecture News, Vol. 33, No. 4, pp. 80–91. DOI: 10.1145/1105734.1105745.
9. **Corporation, N. D. (2019).** Tomoyo linux. tomoyo.sourceforge.net.
10. **Eranian, S., Gourion, E., Moseley, T., Bruijn, W. (2015).** Linux kernel profiling with perf. perf.wiki.kernel.org/index.php/Tutorial.
11. **Garcia-Garcia, A., Saez, J. C., Prieto-Matias, M. (2018).** Contention-aware fair scheduling for asymmetric single-isa multicore systems. IEEE Transactions on Computers, Vol. 67, No. 12, pp. 1703–1719. DOI: 10.1109/tc.2018.2836418.
12. **Goglin, B. (2017).** On the overhead of topology discovery for locality-aware scheduling in HPC. Proceedings of the 25th Euromicro International Conference on Paralel, Distributed and Network-based Processing, pp. 186–190. DOI: 10.1109/pdp.2017.35.
13. **Gouicem, R. (2020).** Thread scheduling in multi-core operating systems: How to understand, improve and fix your scheduler. Ph.D. thesis, Sorbonne Université, France.
14. **Harris, J. A., Cordani, J. (2002).** Schaum's outline of operating systems. Chapter 2: Process Management, McGraw-Hill, pp. 14–23.
15. **Herdrich, A., Illikkal, R., Iyer, R., Newell, D., Chadha, V., Moses, J. (2009).** Rate-based QoS techniques for cache/memory in CMP platforms. Proceedings of the 23th International Conference on Supercomputing, pp. 479–488. DOI: 10.1145/1542275.1542342.
16. **Intel (2007).** Intel®core™2 duo processor e6550. https://ark.intel.com/content/www/us/en/ark/products/30783/intel-core-2-duo-processor-e6550-4m-cache-2-33-ghz-1333-mhz-fsb.html.
17. **Intel (2011).** Intel®64 and IA-32 architectures software developer manuals. https://www.intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html.
18. **John, O. (2001).** Operating systems with linux. Chapter 3: Process Manager, Palgrave, pp. 39–67.
19. **Jones, M. (2008).** Anatomy of linux dynamic libraries. developer.ibm.com/tutorials/l-dynamic-libraries/.
20. **Jung, J., Shin, J., Hong, J., Lee, J., Kuo, T. W. (2017).** A fair scheduling algorithm for multiprocessor systems using a task satisfaction index. Proceedings of the International Conference on Research in Adaptive and Convergent Systems, pp. 269–274. DOI: 10.1145/3129676.3129736.
21. **Lim, G., Min, C. W., Eom, I. Y. (2012).** Load-balancing for improving user responsiveness on multicore

embedded systems. Proceedings of the Linux Symposium, pp. 25–34. DOI: 10.48550/arXiv.2101.09359.

22. **Lozi, J. P., Lepers, B., Funston, J., Gaud, F., Fedorova, A., Quéma, V. (2016).** The linux scheduler: A decade of wasted cores. Proceedings of the Eleventh European Conference on Computer Systems (EuroSys'16), pp. 1–16. DOI: 10.1145/2901318.2901326.
23. **Lozi, J. P., Lepers, B., Funston, J., Gaud, F., Fedorova, A., Quéma, V. (2016).** Your cores are slacking off–or why os scheduling is a hard problem. USENIX Association, Vol. 41, No. 4, pp. 6–13.
24. **Marinakis, T., Haritatos, A. H., Nikas, K., Goumas, G. I., Anagnostopoulos, I. (2017).** An efficient and fair scheduling policy for multiprocessor platforms. pp. 1–4. DOI: 10.1109/ISCAS.2017.8050758.
25. **Mosberger, D., Eranian, S. (2002).** IA-64 linux kernel: Design and implementation. Prentice Hall.
26. **Oprofile (2013).** Oprofile - a system profiler for linux. oprofile.sourceforge.io.
27. **Oracle (2008).** Linkers and library guide. docs.oracle.com/cd/E23824_01/html/819-0690/toc.html.
28. **Pathania, A., Venkataramani, V., Shafique, M., Mitra, T., Henkel, J. (2016).** Distributed fair scheduling for many-cores. pp. 379–384.
29. **Permon2 (2013).** perfmon2 - improving performance monitoring on linux. perfmon2.sourceforge.net.
30. **Sáez, J. C., Gómez, J. I., Prieto, M. (2008).** Improving priority enforcement via non-work-conserving scheduling. Proceedings of the 37th International Conference on Parallel Processing, pp. 99–106. DOI: 10.1109/ICPP.2008.38.
31. **Salzberg-Rodriguez, C., Fischer, G., Smolski, S. (2005).** The Linux®kernel primer: A top-down approach for x86 and powerpc architectures. Chapter 3: Processes: The Principal Model of Execution, Prentice Hall, pp. 77–178.
32. **Shi, Q., Chen, T., Hu, W., Huang, C. (2009).** Load balance scheduling algorithm for CMP architecture. Proceedings of the International Conference on Electronic Computer Technology, pp. 396–400. DOI: 10.1109/icect.2009.74.
33. **Siddha, S., Pallipadi, V., Mallick, A. (2005).** Chip multiprocessing aware linux kernel scheduler. Proceedings of the Linux Symposium, pp. 193–204.
34. **Silberschatz, A., Baer-Galvin, P., Gagne, G. (2007).** Operating system concepts with java. John Wiley and Sons.
35. **Silberschatz, A., Baer-Galvin, P., Gagne, G. (2018).** Operating system concepts. Chapter 5: CPU Scheduling, John Wiley and Sons, pp. 220–227.
36. **SPEC (2007).** Standard performance evaluation corporation. www.spec.org.
37. **Tanenbaum, A. S., Herbert, B. (2015).** Modern operating systems. Chapter 8: Multiple Processor Systems, Person Education, pp. 520–539.
38. **Tiwari, V., Malik, S., Wolfe, A., Lee, M. T. C. (1996).** Instruction level power analysis and optimization of software. Proceedings of 9th International Conference on VLSI Design, pp. 1–18. DOI: 10.1109/icvd.1996.489624.
39. **Vogl, S., Eckert, C. (2012).** Using hardware performance events for instruction-level monitoring on the x86. Proceedings of EuroSec'12, 5th European Workshop on System Security.
40. **Weaver, V. (2013).** Linux perf_event features and overhead. FastPath: Second International Workshop on Performance Analisys of Workload Optimized Systems, pp. 1–6.
41. **Weaver, V. (2013).** perf_event – programming guide. web.eece.maine.edu/~vweaver/projects/perf_events/programming.html.

42. Wolfgang, M. (2008). Professional Linux®kernel architecture. Wiley Publishing.

43. Zhang, X., Zhong, R., Dwarkadas, S., Shen, K. (2012). A flexible framework for throttling-enabled multicore managment. Proceedings of the 41st International Conference on Parallel Processing, pp. 389–398.

44. Zhuravlev, S., Blagodurov, S., Fedorova, A. (2010). Addressing shared resource contention in multicore processors via scheduling. ACM SIGARCH Computer Architecture News, Vol. 38, No. 1, pp. 129–142. DOI: 10.1145/1735970.1736036.

Article received on 25/11/2022; accepted on 12/06/2024.
** Corresponding author is Marco A. Ramírez-Salinas.*

ISSN 2007-9737

Rocket Thrust Vectoring Attitude Control based on Convolutional Neural Networks

Rodolfo Garcia-Rodriguez*, Ivan Martinez-Perez, Luis E. Ramos-Velasco, Mario A. Vega-Navarrete[1]

Universidad Politécnica Metropolitana de Hidalgo,
Postgraduate Program in Aerospacial Engineering,
Mexico

{rogarcia, lramos, mvega}@upmh.edu.mx

Abstract. Launching and landing rockets on the Earth and in space have had intensive research and development in the last years. The idea of reducing costs is related regularly to the reusability of some mission stages. Though the launch has attracted attention due to including the main engines fundamental to boosting spacecraft onto an orbital or interplanetary trajectory, the landing takes relevance in space missions. While the rocket's landing has been carried out on Earth using an autonomous spaceport drone ship, it is challenging to design intelligent model-free systems that can continuously learn and compensate for slight deviations until they meet the target. This paper focuses on studying vertical rocket landing using convolutional neural networks. Assuming that the rocket is near the landing area, an attitude rocket control is proposed using a vision system to recognize it and drive -the nozzle TVC. Experimental results show the attitude control commanded by a nozzle TVC of an experimental rocket under different conditions.

Keywords. Rocket thrust vector control, convolutional neural networks, attitude rocket control.

1 Introduction

In the last years, there has been an increased need to reduce the costs in the space launches through spacecraft reusable. In this way, many companies, like the US company Space X, have focused their efforts on the vertical landing where the rocket's first stage should be reusable because it is higher cost.

In particular, the Vertical Takeoff Vertical Landing, VTVL, technology has been used successfully recently by companies as SpaceX (Falcon 9), Blue Origin (New Shepard) for reusable rockets. However, although VTLV has been demonstrated to be helpful, some open problems remain open in vertical takeoff technology and reusable landing rockets.

Since the beginning, the rocket's landing has attracted attention on how to get precision landing on the Earth and a more challenge in space where the parachute does not seem the best option [1].

In the literature, the rocket landing has been studied in two ways: a) as the reentry problem in the atmosphere where the high acceleration, dynamic pressure, and heating rate represents constraints of the optimization problem where the aerodynamic drag is fundamental to get deceleration effect and generates the reference rocket trajectory, and b) vertical landing problem where the altitude and speed of the rocket are controlling adjusting engine trust in a vacuum environment [15].

Due to the near of the landing area, the aerodynamic forces are in equilibrium with the gravity force; the remaining mass of the rocket defines the landing velocity. Thus, the rocket's altitude, speed, and attitude are adjusted to satisfy the landing time. This paper focuses on the vertical rocket landing on a floating landing platform known as the Autonomous Spaceport Droneship (ASDS), where precision is highly challenging.

Mainly to increase the rocket landing precision and divert or move sideways, the vectored thrust, TVC, is considered [3, 10, 6]. Finally, due to the

Table 1. Image augmentation parameters

Train_datagen / Evaluate_datagen	
Rescale	1./255
Rotation_range	40,
Width_shift_range	0.2
Height_shift_range	0.2
Shear_range	0.2
Zoom_range	0.2
Horizontal_flip	True

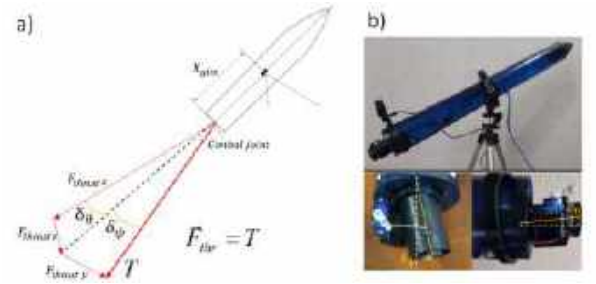


Fig. 1. (a) A TVC on the rocket, (b) Experimental prototype

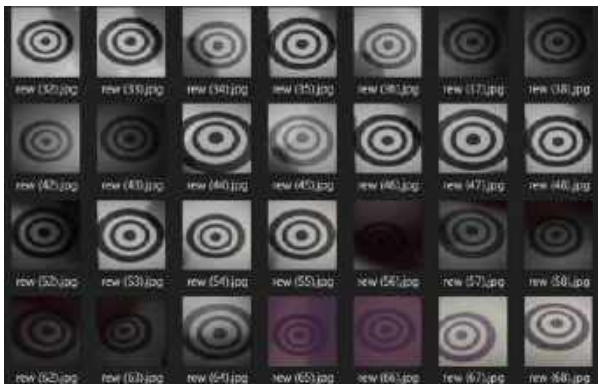


Fig. 2. Images from the landing platform dataset

landing precision capabilities vary on the Earth and space, this problem implies the researches and developments of many fields as structural dynamics, engines, control systems taken place simultaneously. The rocket landing problem is studied and solved as an optimization problem under different constraints where the goal is to generate online flight trajectory [12, 8, 2].

The principal drawback of these approaches is the lack of knowledge of some parameters, aerodynamics or physics, in certain conditions, and robustness to compensate the parameters uncertainty, the wind perturbations, or noises that can affect the rocket's landing.

Recently, advances in machine learning tools, like deep learning, have been used to solve some problems in aircraft or rockets where the principal goal is a parameter estimation from the available data and improve the capacity of parameters estimation through a learning algorithm. The learning algorithms aim to estimate an unknown mapping from available data to predict future data, commonly known as generalization.

Deep Learning is part of Machine Learning, where information processing is carried out in hierarchical layers. That is, the deep learning algorithms are based on neural networks [14]. In addition to the traditional neural network approximating the unknown mapping from the input-output data, the deep learning algorithms can learn basic features from the input data in a deep learning algorithm.

They have the capacity to understand some characteristics of the new data that are useful to make predictions. The interesting feature of deep neural networks is that the learning capacity is related to hidden layers number [7]. Some approaches have been developed taking advantage of the deep learning algorithms as in [13] where deep neural networks are used to obtain the Hamilton-Jacobi-Bellman solution to guarantee aircraft landing, lunar landing in [5], and rocket launching in [17].

Furthermore, in [16] a deep convolutional neural network is used to process and classify a huge number of images data generated by optimal equipment. This paper uses a deep learning algorithm called Convolutional Neural Network, CNN, to learn and identify the landing platform to control attitude rocket.

The CNN is a feedforward neural network which by convolutional layers applied filters to the input data to obtain features from it, called feature learning. The application of CNN has

Table 2. Fit generator parameters

Training configuration using ImageDataGenerator	
train datagen.flow from directory	target size=(128,128) / batch size=64 /class mode='binary'
validation datagen. flow from directory	target size=(128,128) / batch size=64 / class mode='binary'
model.fit generator	train generator / epochs=30 / steps per epoch=63 / validation data=validation generator / validation steps=7 / workers=4

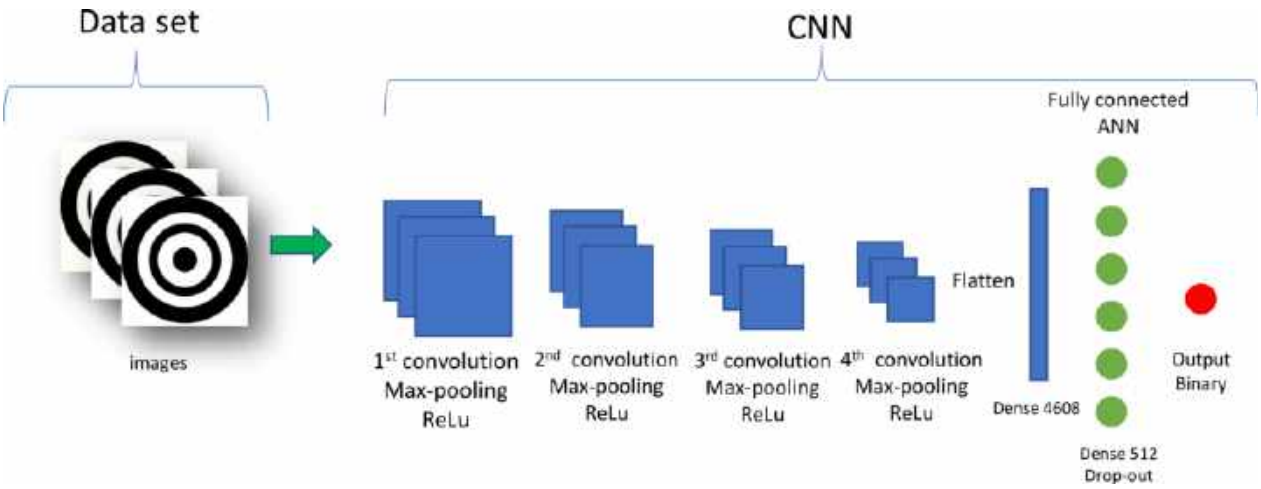


Fig. 3. Architecture of the CNN

been extended recently in many areas as computer vision, speech recognition, machine translation, to name a few. Motivating from concerns mentioned above, the vertical rocket landing, and the learning capabilities of the CNN, the problem to solve is defined as follows.

1.1 Problem Statement

Although the rocket has many phases of the return-to-launch-site mission, we focus on the vertical landing problem, specifically on the rocket's orientation near the landing area. Due to the aircraft does not have a straight path to its destination.

On the contrary, regularly, there is a slight deviation concerning the angle route or trajectory. The problem stands for the attitude rocket control avoiding any knowledge system.

Thus, to get autonomous and intelligent space landings, a vision system will drive a nozzle TVC to guarantee the attitude rocket control while the CNN is trained to recognize the landing platform.

Assuming that the landing platform is moving and the rocket is fixed in its center of gravity, in this paper, a landing platform dataset is built, and the CNN is trained to identify it.

Once a vision system identifies a target in its field of view, FOV, a bounding box is created around it while simultaneously, the coordinates of the center of the landing platform are obtained.

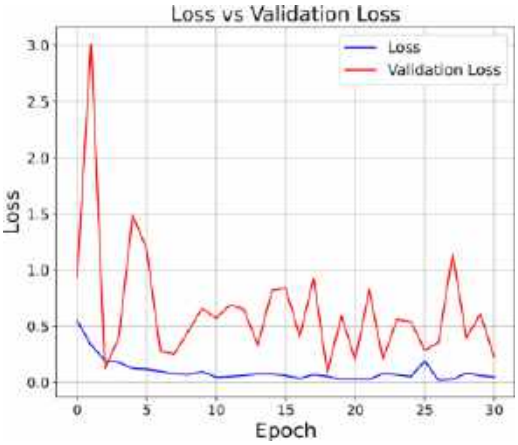


Fig. 4. Loss curve (1484 images)

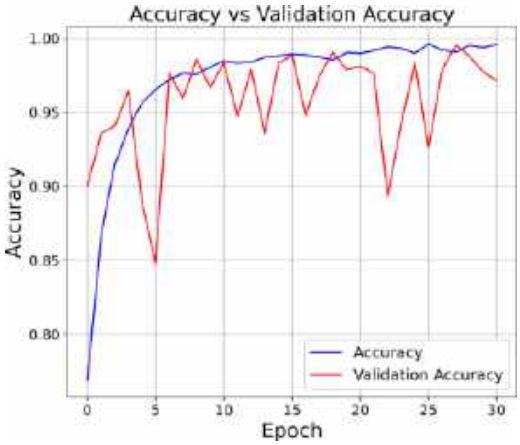


Fig. 5. Accuracy curve (1484 images)

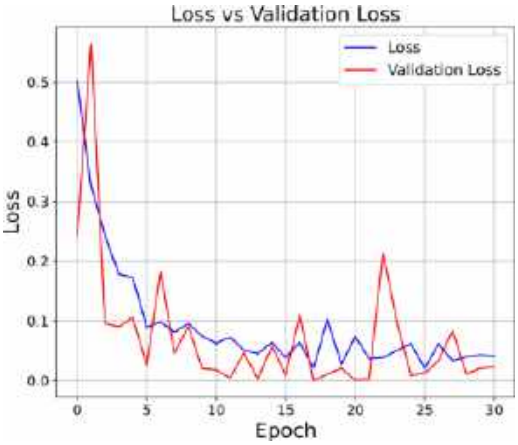


Fig. 6. Loss curve (8 feature extraction layers)

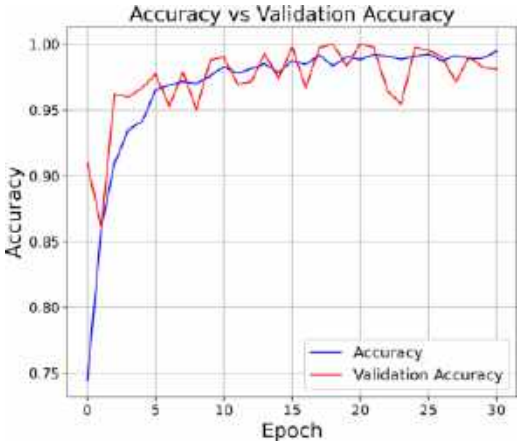


Fig. 7. Accuracy curve (8 feature extraction layers)

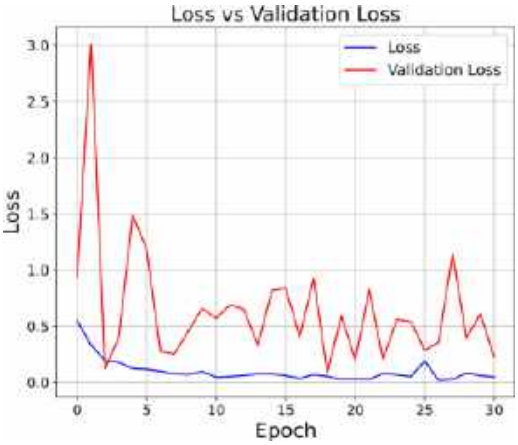


Fig. 8. Loss curve (742 images)

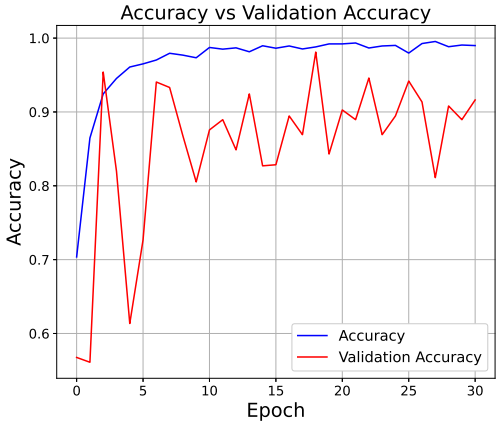


Fig. 9. Accuracy curve (742 images)

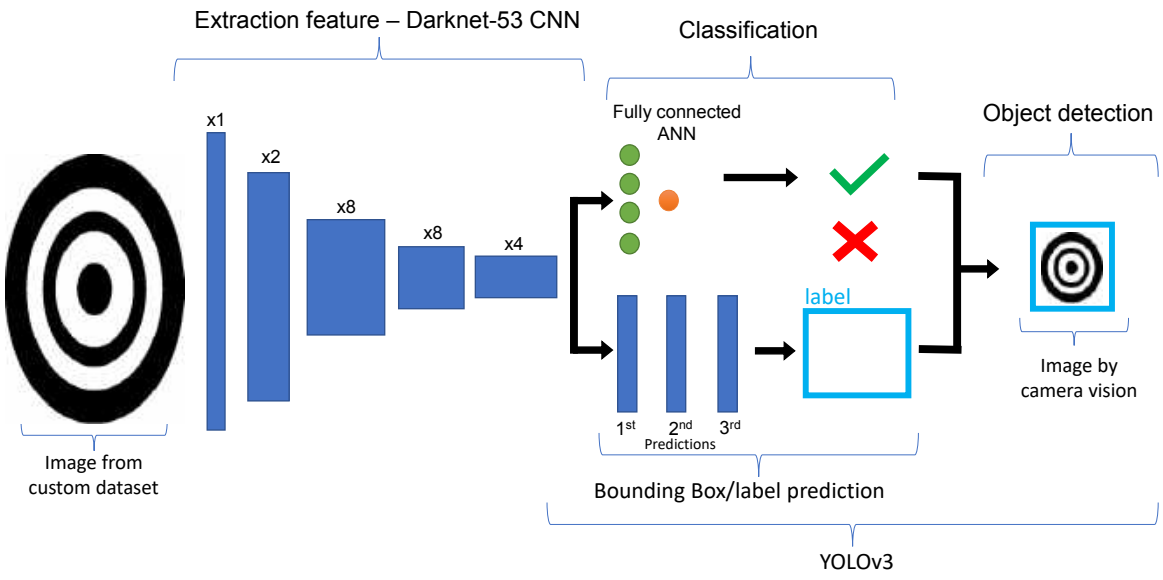
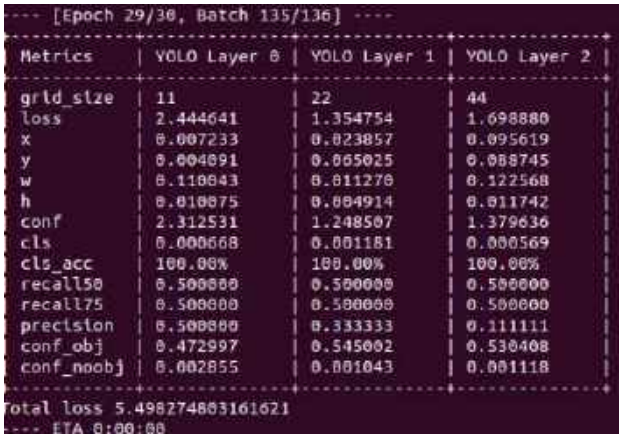


Fig. 10. Landing platform detection algorithm using Yolov3



```
---- [Epoch 29/30, Batch 135/136] ----
+------------+--------------+--------------+--------------+
| Metrics    | YOLO Layer 0 | YOLO Layer 1 | YOLO Layer 2 |
+------------+--------------+--------------+--------------+
| grid_size  | 11           | 22           | 44           |
| loss       | 2.444641     | 1.354754     | 1.698880     |
| x          | 0.007233     | 0.023857     | 0.095619     |
| y          | 0.004891     | 0.005025     | 0.008745     |
| w          | 0.110843     | 0.011270     | 0.122568     |
| h          | 0.010875     | 0.004914     | 0.011742     |
| conf       | 2.312531     | 1.248507     | 1.379636     |
| cls        | 0.000668     | 0.001181     | 0.000569     |
| cls_acc    | 100.00%      | 100.00%      | 100.00%      |
| recall50   | 0.500000     | 0.500000     | 0.500000     |
| recall75   | 0.500000     | 0.500000     | 0.500000     |
| precision  | 0.500000     | 0.333333     | 0.111111     |
| conf_obj   | 0.472997     | 0.545002     | 0.530408     |
| conf_noobj | 0.002855     | 0.001043     | 0.001118     |
+------------+--------------+--------------+--------------+
Total loss 5.498274803161621
---- ETA 0:00:00
```

Fig. 11. Object detection: Training results

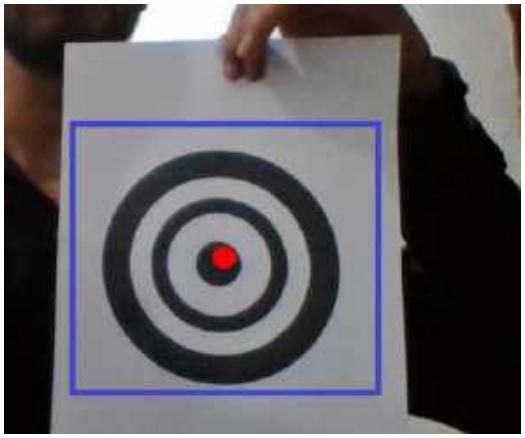


Fig. 12. Landing platform detection on the FOV

Finally, the orientation desired angles are defined, taking as reference the FOV of the vision system, which nozzle TVC uses to guide the appropriate rocket.

Specifically, the nozzle TVC system used on the rocket prototype has two orientations commanded by electromechanical actuators each one, see Fig. 1.

The propulsion used is constant, where a portable air compressor pump supplies it.

The remainder of this paper is organized as follows.

Section 2 presents the description of training and testing of the CNN to get an attitude rocket control by vision system.

Experimental results are presented under different conditions in Section 3.

Finally, some conclusions are presented in Section 4.

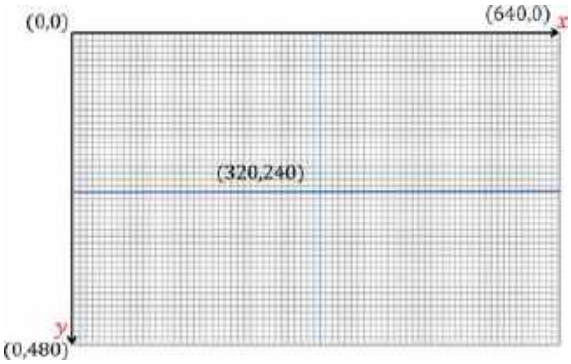


Fig. 13. Reference systems of FOV

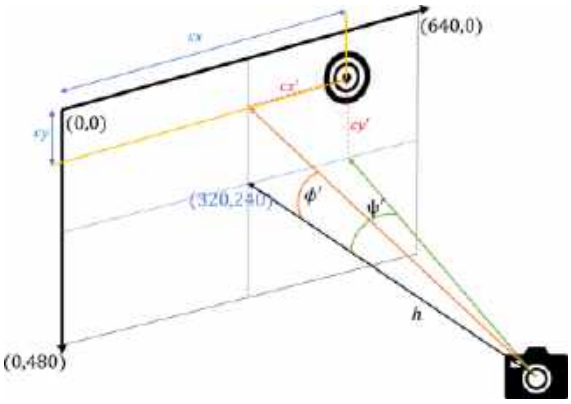


Fig. 14. Obtaining desired angles from FOV

Fig. 15. Experimental prototype

2 Approach

This section presents the different stages to control the rocket attitude using CNN as part of its landing back mission.

Stages as training of CNN, landing platform recognition by the vision system, and the generation of the orientation desired angles are developed in Anaconda Python distribution under Windows©10 operating system. The principal packages installed are Tensorflow 1.13.1, Keras 2.3.1, and OpenCV-python 4.4.0.

2.1 Training of CNN

The first step before CNN training is to build a landing platform dataset.

2.1.1 Landing Platform Database

The landing platform dataset was built using a mobile phone with a resolution of 25 MP. The dataset images have the same size, are labeled, and within them, the landing platform is not centered. The landing platform database has 742 images, where 372 are positive images with different degrees of illumination and photo angles to make the learning and detection more robust, while 370 negative images of anything; other than the target. In Fig. 2 we can see a sample of the database created. Once the landing platform has been built, the next step is to train the CNN to recognize the landing platform and use it to rocket control orientation.

2.1.2 Convolutional Neural Network

The training goal of the CNN is to get a valuable set of weights that allow recognizing the landing platform and the generalization ability of the model. Finally, the set of weights is saved for testing the learning. Before training the CNN, the image augmentation technique is applied to the dataset images to get multiple transformed copies of an image.

The image augmentation technique will be helpful to generalize the model due to adding variations levels of unseen data and avoiding model overfitting. The Image data generator is a class of Keras that configure and convert the database into useful data to be entered in CNN. Finally, each new batch of our data is adjusted randomly.

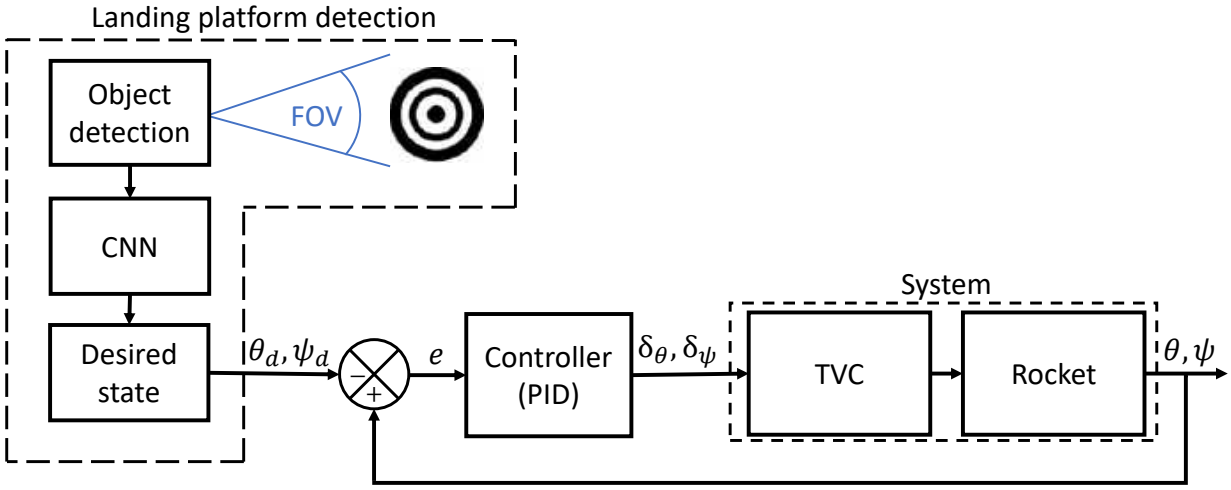


Fig. 16. Rocket thrust vectoring control system using convolutional neural network, CNN

In Table 1 are shown the augmentation parameters apply to the dataset images while parameters of the Flow_from_dataframe and Keras Fit_generator method are shown in Table 2. The former directly augment images by reading its name and target value from a dataset, and the latter accepts a batch of the dataset updates the model's weights.

Now, we are ready to train our CNN. The CNN uses an image 224 $\times$ 224 pixels, four feature extraction layers, and two fully connected neural networks in the output layer, see Fig. 3. The training model was carried out using 30 epochs with a batch size set to 64.

The optimization algorithm used was the RMSprop with a learning rate set to 0.0003. The optimization algorithm used was the RMSprop with a learning rate set to 0.0003. Finally, the results of the training using augmenting images database are given in Fig. 8 and Fig. 9 with the loss and accuracy curves, respectively.

It is observed from Fig. 8 that the training loss curve moves down until it reaches a value of approximately 0.1, while validation loss has an expected downward trend but in some epochs display peaks. These peaks represent the images that the model never saw and cannot recognize.

On the other hand, the training and validation accuracy curves, Fig. 9, tend to increase which means the training is adequate even to the small number of epochs used for the training. Thus, the neural network has trained to generalize the images from loss and accuracy curves.

Remark. Fig. 4 and 5 show the training and validation results using 1,848 images. The performance is better than in Fig. 8-9 but appears some peaks that represent that there are images that cannot be learned for the CNN. By another hand, if the feature extraction layers are increased to eight, the performance of the training and validation significantly improves, see Fig. 6-7. Thus, the features extraction layers are fundamental in the learning process to predict future data accuracy. Once the CNN is trained, the next step is to landing platform recognition using the vision system.

2.2 Rocket Thrust Vectoring Control Using CNN

2.2.1 Landing Platform Recognition

Learning an object or image using CNN is generally used for classification or categorization tasks where the object recognized within an image is fundamental.

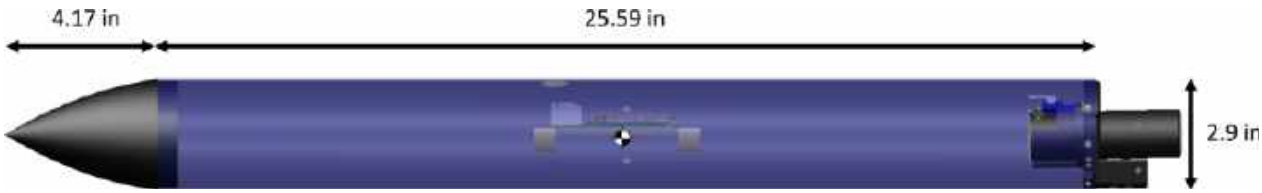


Fig. 17. Experimental rocket dimensions

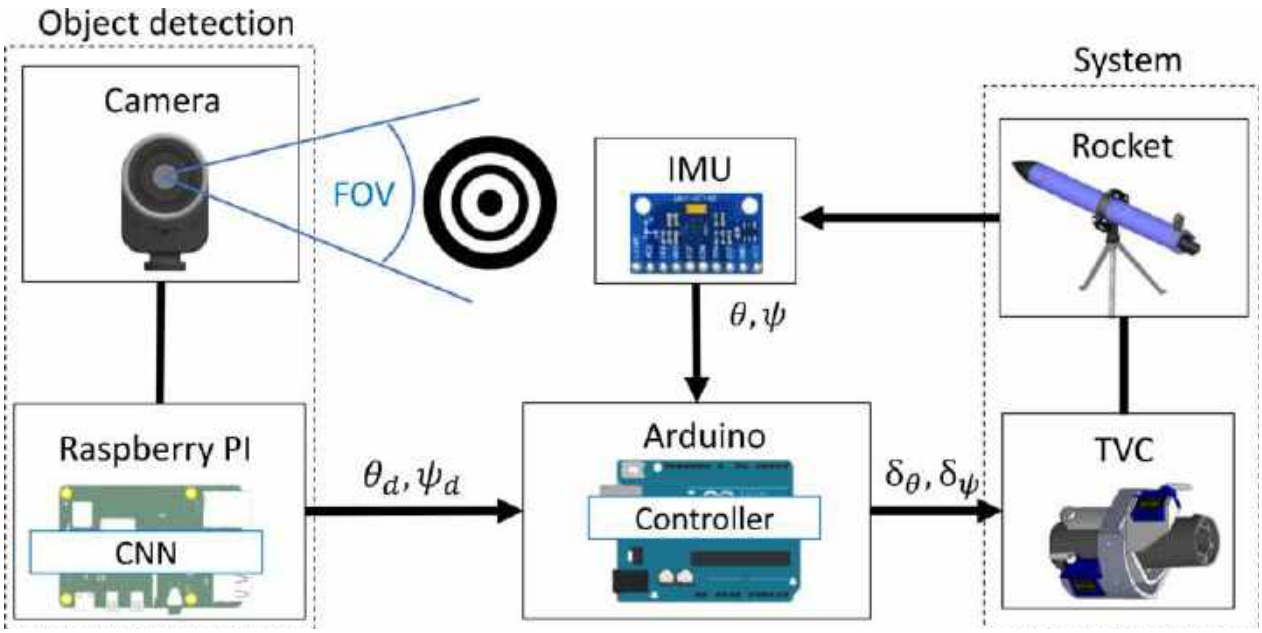


Fig. 18. General scheme used to generate the TVC signals

In this paper, we are interested, in addition, to knowing the exact location of the landing platform, see Fig. 12. Thus, the images from the dataset were labeled with coordinates of the object's location embedded into the images. To creates the labels is used the labelImg application. It is a graphical image labeling application tool written in Python where the coordinates are manually saved in a TXT file for each image.

Finally, in conjunction with labels, this dataset is trained to obtain its location in the camera's field of view. To get the object detection[1] real-time object detection algorithm named Yolov3 is used where a Darknet-53 CNN is in charge of learning and classifying the desired image, Fig. 10.

Thus, simultaneously the bounding box regressor[2] and the label's prediction are made in the case of having several classes (different objects/images) [4, 9, 11].

The bounding box uses residual blocks algorithms for better performance in tiny images, also Intersection Over Union, IOU, techniques to get a perfect box over the object. In addition, it uses three prediction scales to get better performance, while the labeling uses binary cross-entropy loss because there are many overlapping labels. Given that CNN complete training used in YoloV3 takes much computational time and requires significant computational power, a transfer learning technique is used to reduce neural network training time.

[1]The process of recognizing and getting its location is called object detection.

[2]Box that marks the target's location (landing platform) on the image.

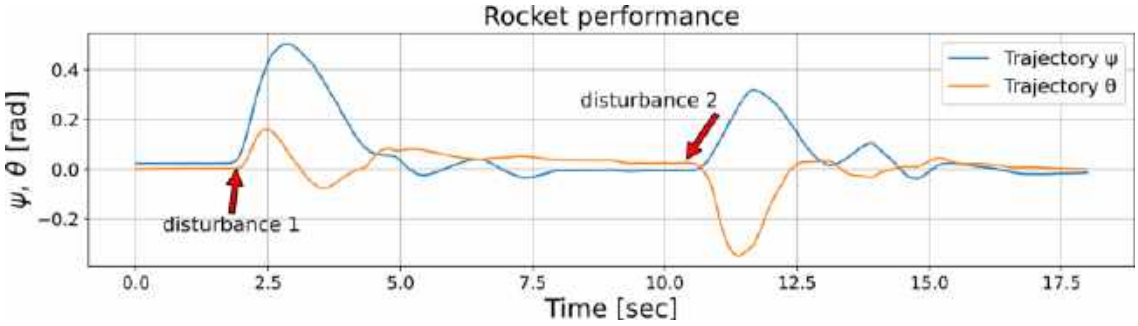


Fig. 19. Experiment 1: Performance of θ and ψ angles under different perturbations

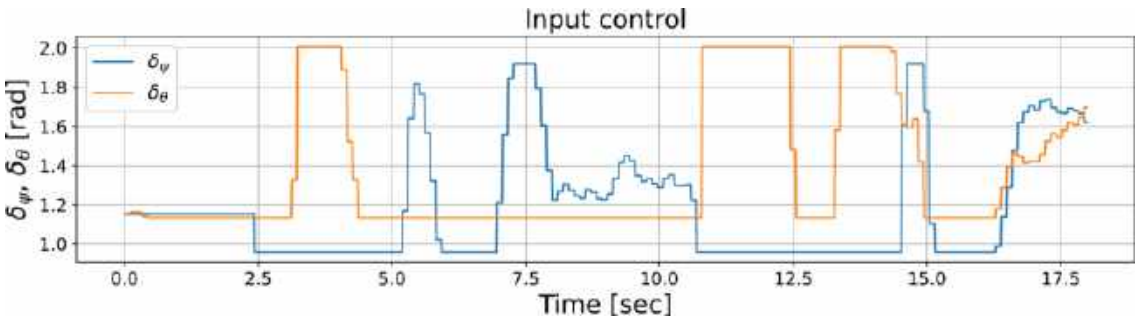


Fig. 20. Experiment 1: Control signals:δ_θ and δ_ψ

Thus, using a pre-trained CNN that contains the information to recognize faster generic features from any image, the last three layers of the CNN are trained to recognize a particular object, in this case, the landing platform.

That is, the pre-trained weights remain fixed while the training takes place. The training model parameters used for the complete training of the object detection were 30 epochs with a batch size set to 8. The optimization algorithm used was the MSEloss. Fig. 11 show the training results.

2.2.2 Generating the Orientation Desired Angles

Once the artificial vision system recognizes the landing platform and its location, it is possible to calculate the desired orientation of the rocket angles. Let the coordinates of the field of view be defined according to Fig. 13, where the origin is placed in the upper left corner, the horizontal axis has 640 pixels, and the vertical axis has 480 pixels.

Thus, using the landing platform coordinates is possible to calculate the angles concerning each of the axes of the field of view, see Fig. 14. Such that θ_d and ψ_d are given as:

$$\theta_d = \tan^{-1}\left(\frac{cy'}{h}\right) = \tan^{-1}\left(\frac{cy - 480/2}{h}\right), \quad (1)$$

$$\psi_d = \tan^{-1}\left(\frac{cx'}{h}\right) = \tan^{-1}\left(\frac{cx - 640/2}{h}\right), \quad (2)$$

where $(cx,\ cy)$ are the coordinates from the origin (0,0) to the landing platform detected, $(cx',\ cy')$ are the coordinates from the median axes, with origin in (320, 240), to the landing platform, and h is the distance from the rocket's camera to the target where for this application is assumed constant.

Notice that the angles concerning each of the axes of the field of view, (1)-(2), are no more than the desired angles to be used rocket attitude control.

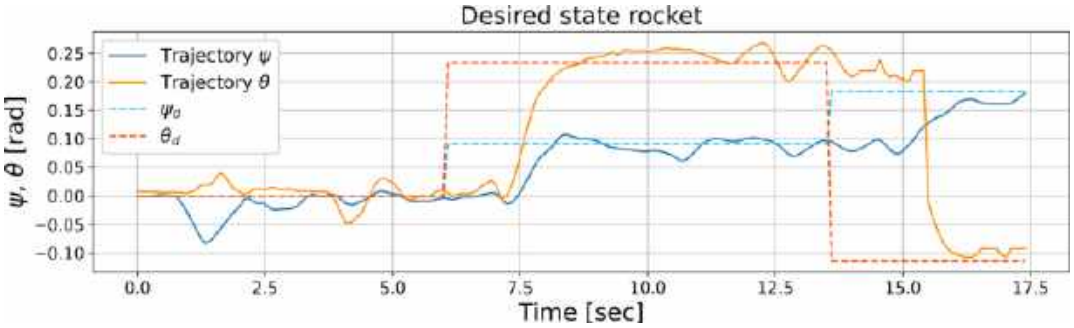


Fig. 21. Experiment 2: Desired and actual rocket angles

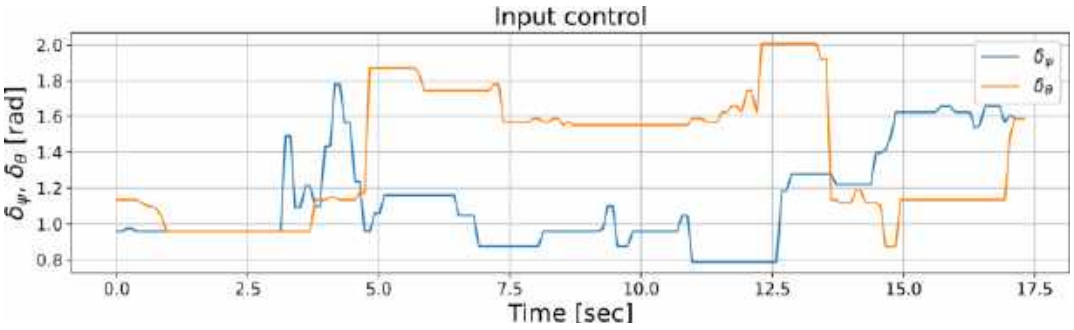


Fig. 22. Experiment 2: Control signals: δ_θ and δ_ψ

2.2.3 Rocket Thrust Vectoring Control

Finally, once the desired angles θ_d and ψ_d are defined, we can rocket attitude control using nozzle TVC. The general control scheme proposed is shown in Fig. 16. For simplicity in this paper, we consider a traditional PID controller on a process variable (PV) and a setpoint in discrete form defined as:

$$\delta_*(t_k) = P + I + D, \tag{3}$$

where:

$P = K_P\, e(t_k)$,

$I = K_I \sum_{i=1}^{k} e_i(t_k)\Delta t_k$,

$D = K_D \dfrac{\Delta e(t_k)}{\Delta t_k}$,

where K_P, K_I, K_D are the feedback gains, $e(t_k) = *(t_k) - *(t_K)_d$ is the tracking error at time step t_k with $*(t_k)$ the actual state and, $*(t_k)_d$ the desired state, and Δt_k is the step size. In this case the PID controller is utilized to control the states θ, and ψ. Finally, Fig. 15 shows the prototype model rocket used for experiments. More technical details about the prototype is given in the next section.

3 Experimental Results

3.1 Experimental Setup

To demonstrate the performance of the proposed scheme an experimental results were carried out on a fixed at the center of gravity rocket, see Fig. 15. The rocket comprises a nose cone, body tube, and TVC nozzle with two degrees of freedom.

The experimental rocket principal dimensions are shown in Fig. 17 where the fineness of the nose is 1.39, and its approximated total weight is 615 g. As the rocket is fixed at the center of gravity, as propulsion is used a portable air compressor pump with pressure around 300 KPa, obtaining a thrust of 6.2×10^{-3} kN.

To identify the landing platform, on the rocket is placed the vision system through a web camera, Genius Qcam 6000 of 2MP. In addition, the OpenCV library is installed on a Raspberry Pi 4, 8Gb RAM, located inside the body tube to recognize the landing platform. Finally, CNN is programming on Raspberry Pi 4, too. On the other hand, an IMU MPU-6050 is used to carry out the rocket orientation measurement: θ, and ψ angles. Thus, once the actual angles and the coordinates landing platform are available, the nozzle TVC control signals are calculated on Arduino UNO, also inside the body tube. See Fig. 18.

Remark. The desired angles information in sent from the Raspberry Pi to Arduino through SERIAL communication (SSH). Although this communication kind produce a delay, this communication has a type of switch which causes the sent data to pile up, the attitude control is carried out adequately. Due to data Arduino arriving in a non-existent format, a delay of 3 sec. is programmed on the Raspberry.

3.2 Experimental Conditions

The objective of the experiments is to rocket thrust vectoring control orientation; that is, the rocket should be oriented according to the landing platform that is assumed is moving.

In the first experiment, the goal is to test that the TVC nozzle can maintain the rocket in equilibrium positions under different perturbations, that is, $\theta = \psi \approx 0$. Fig. 19 shows the performance of the θ and ψ angles when the rocket is subject to two manual perturbations. Notice that, after the induced perturbations, the TVC nozzle can stabilize the rocket, that is, $\theta = \psi \to 0$.

In Fig. 20 is shows the control signals applied to the TVC nozzle. In the second experiment, the vision system on the rocket is used to recognize and follow the landing platform using a TVC nozzle. The CNN is used to acknowledge and calculate the landing platform coordinates representing the desired angles, θ_d, and ψ_d.

Thus, the controller's goal of the TVC nozzle is to generate the control signals to get the desired rocket orientation. Fig. 21 shows how once the desired angles are generated, the controller produces the adequate control signals applied to the TVC that guarantee to reach the desired angles. Notice that exists a delay of around 7 s between each sending of coordinates. This is due to delay communication between Arduino and Raspberry. Therefore, the rocket reaches the desired angles despite this delay and noise on orientations measurements. Finally, in Fig. 22 the control signals performance of θ_d and ψ_d are show.

4 Conclusion and Future Work

The vertical rocket landing using CNN, especially the attitude control by nozzle TVC without knowing the system or landing trajectory, is presented. Assuming that the landing platform is moving, the rocket's orientation is driven by a vision system that recognizes the target, controlling the rocket orientation by the nozzle TVC. In addition, the building dataset, training, and testing of the CNN is presented. Although experimental results are done under a controlled environment, the rocket is fixed on the center of gravity. The results show that the proposed scheme is robust enough to latency, noise in sensors, and motors, to control rocket orientation using CNN. In this way, the CNN can be considered part of an intelligent system for launching and landing tasks where the learning characteristics from new data to make predictions without any knowledge of the system or desired trajectory can be helpful from space missions.

Acknowledgments

This work was partially supported by CONACYT Scholarships with reference number 0055803.

References

1. **Blackmore, L. (2016).** Autonomous precision landing of space rockets. Frontiers of Engineering, Vol. 46, No. 4, pp. 15–20.

2. **Blackmore, L., Açikmeşe, B., Scharf, D. P. (2010).** Minimum-landing-error powered-descent guidance for mars landing using convex optimization. Journal of Guidance, Control, and Dynamics, Vol. 33, No. 4, pp. 1161–1171. DOI: 10.2514/1.47202.
3. **Devlin, T., Dickerhoff, R., Durney, K., Forrest, A., Pansodtee, P., Adabi, A., Teodorescu, M. (2018).** ElbowQuad: Thrust vectoring quadcopter. AIAA Information Systems-AIAA Infotech @ Aerospace, pp. 8–12. DOI: 10.2514/6.2018-0893.
4. **Farhadi, A., Redmon, J. (2018).** Yolov3: An incremental improvement. Computer Vision and Pattern Recognition, pp. 1804–2767. DOI: 10.48550/ARXIV.1804.02767.
5. **Furfaro, R., Bloise, I., Orlandelli, M., Di-Lizia, P., Topputo, F., Linares, R. (2018).** Deep learning for autonomous lunar landing. Master's thesis, Politecnico di Milano and University of Arizona.
6. **Knuth, D. E. (1992).** Literate programming. , No. 27, pp. 97–111.
7. **Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012).** Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, Curran Associates, Inc., Vol. 25.
8. **Liu, X., Shen, Z., Lu, P. (2016).** Entry trajectory optimization by second-order cone programming. Journal of Guidance, Control, and Dynamics, Vol. 39, No. 2, pp. 227–241. DOI: 10.2514/1.g001210.
9. **Mao, Q. C., Sun, H. M., Liu, Y. B., Jia, R. S. (2019).** Mini-YOLOv3: Real-time object detector for embedded applications. IEEE Access, Vol. 7, pp. 133529–133538. DOI: 10.1109/access.2019.2941547.
10. **Oates, G. C. (1984).** Aerothermodynamics of gas turbine and rocket propulsion. American Institute of Aeronautics and Astronautics. DOI: 10.2514/4.861345.
11. **Redmon, J., Divvala, S., Girshick, R., Farhadi, A. (2016).** You only look once: Unified, real-time object detection. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788. DOI: 10.48550/arXiv.1506.02640.
12. **Sagliano, M., Mooij, E., Theil, S. (2017).** Onboard trajectory generation for entry vehicles via adaptive multivariate pseudospectral interpolation. Journal of Guidance, Control, and Dynamics, Vol. 40, No. 2, pp. 466–476. DOI: 10.2514/1.G001817.
13. **Sánchez-Sánchez, C., Izzo, D. (2018).** Real-time optimal control via deep neural networks: Study on landing problems. Journal of Guidance, Control, and Dynamics, Vol. 41, No. 5, pp. 1122–1135. DOI: 10.2514/1.G002357.
14. **Vasilev, I., Slater, D., Spacagna, G., Roelants, P., Zocca, V. (2019).** Python deep learning: Exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow. Packt Publishing.
15. **Wang, C., Song, Z. (2019).** Trajectory optimization for reusable rocket landing. Chinese Automation Congress, pp. 3052–3057. DOI: 10.1109/CAC48633.2019.8997476.
16. **Zhang, L., Chen, Z., Wang, J., Huang, Z. (2018).** Rocket image classification based on deep convolutional neural network. 10th International Conference on Communications, Circuits and Systems, pp. 383–386. DOI: 10.1109/ICCCAS.2018.8769176.
17. **Zhou, G., Fan, Y., Cui, R., Bian, W., Zhu, X., Gai, K. (2018).** Rocket launching: A universal and efficient framework for training well-performing light net. DOI: 10.48550/arXiv.1708.04106.

Article received on 30/08/2022; accepted on 15/04/2024.
** Corresponding author is Rodolfo Garcia-Rodriguez.*

ISSN 2007-9737

Applying Support Vector Machines with Different Kernel to Breast Cancer Diagnosis

Seyyid Ahmed-Medjahed[1,*], Fatima Boukhatem[2]

[1] University of Relizane, Relizane, Algeria

[2] University of Djillali Liabes, Sidi Bel Abbes, Algeria

seyyidahmed.medjahed@univ-relizane.dz, fatima.boukhatem@univ-sba.dz

Abstract. The detection of breast cancer poses a significant challenge in the field of medicine. It represents the second type of the largest cases of cancer deaths in women. Several techniques have been found to solve the problem or make a better diagnosis. Recently, Support Vector Machine based systems are the most common and are considered a better diagnostic assistant in cancer detection research. The quality of the results generated depends on the choice of some parameters such as the kernel function and the model parameters. In this paper, we analyze and evaluate the performance of several kernel functions in the SVM algorithm. Experiments are conducted with different training-test phases generated by the holdout method and we used the WBCD (Wisconsin Breast Cancer Database) to analyze the results. The results are evaluated by using the following performances measures: classification accuracy rate, sensitivity, specificity, positive and negative predictive values. To validate the results obtained by these different kernel functions, we use different values for the kernel functions parameters and SVM model parameters and we record the optimal parameters values. Finally, we show that the Cauchy kernel and the Rational Quadratic kernel are identical and converge to the same value.

Keywords. Support vector machine, kernel function, breast cancer, diagnosis, classification, sequential minimal optimization.

1 Introduction

Breast cancer originates from the inner lining of milk ducts or the lobules responsible for supplying milk to the ducts. It manifests as a tumor within the breast, which can either be benign (non-cancerous) or malignant (cancerous).

Malignant tumors grow and develop into cancer. Breast cancer is a leading cause of mortality worldwide. Every year the breast cancer is detected in 1,3 million women on the world, the mortality rate increases rapidly with more than 1,6 million cases in 2010 which corresponds to 425000 deaths [6]. The correct diagnosis of breast cancer is very important to help the doctors.

Detecting cancerous cells at an early stage, before they spread, can significantly increase the survival rate for patients by over 97%[1] Nevertheless, classifier systems are widely used to solve the problem of cancer classification and to help the experts to make a good diagnosis. The major advantages of classifier systems are the minimization of possible errors that might be made and the ability to give a detailed examination.

The Support Vector Machine (SVM) is recognized as one of the most widely used classifier systems which has become quite an active research field in machine learning. Recently, Support Vector Machines have been shown to give good results and a good generalization performance in the medical diagnosis field precisely in cancer classification [19, 8, 11, 3, 17].

[1] American Cancer Society Hompage. (2008). Citing Internet sources Available from: www.cancer.org.

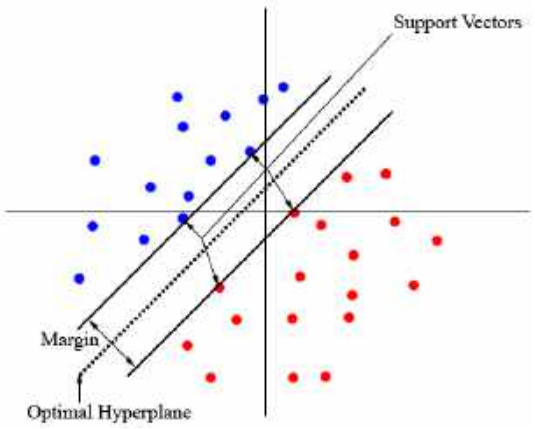


Fig. 1. The figure shows the optimal hyperplane and the support vectors

Table 1. Some kernel functions defined in the literature (M. is multiquadric and q. is quadratic)

Kernel name	Formulation
Linear	$k(x,y) = x^t.y$
Polynomial	$k(x,y) = (x^t.y)^d$
Gaussian	$k(x,y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$
Sigmoid	$k(x,y) = \tan(P1.x^t.y + P2)$
Cauchy	$k(x,y) = \frac{1}{1+\frac{\|x-y\|^2}{\sigma}}$
Inverse M.	$k(x,y) = \frac{1}{\sqrt{\|x-y\|^2+c^2}}$
Quadratic	$k(x,y) = (x^t.y+1)^2$
Multiquadric	$k(x,y) = \sqrt{\|x-y\|^2 + c^2}$
Power	$k(x,y) = -\|x-y\|^d$
Rational Q.	$k(x,y) = 1 - \frac{\|x-y\|2}{\|x-y\|2+c}$
Wave	$k(x,y) = \frac{\theta}{\|x-y\|}\sin\frac{\|x-y\|}{\theta}$
Spherical	$k(x,y) = 1 - \frac{3}{2}\frac{\|x-y\|}{\sigma} + \frac{1}{2}(\frac{\|x-y\|}{\sigma})^3$

Achieving strong performance with the SVM method heavily relies on the selection of appropriate kernel functions. These functions enable the algorithm to identify the maximum-margin hyperplane within a transformed feature space.

Additionally, the effectiveness of the SVM hinges on the careful tuning of kernel function parameters and the cost parameter C in the SVM model has a very important role to get a good classification accuracy rate [18]. Numerous studies in the literature have explored medical diagnosis of breast cancer using the Wisconsin Breast Cancer Database (WBCD).

For instance, Quinlan achieved a classification accuracy rate of 94,74% through $10-$fold cross-validation using the C4.5 decision tree method [16]. Similarly, Hamilton et al. achieved a classification accuracy rate of 95,00% using the RIAC method [9].

Nauck and Kruse obtained 95,06% classification accuracy rate by using the neuron-fuzzy techniques [13]. Albrecht et al. reached 98,80% classification accuracy rate with logarithmic simulated annealing with the perceptron algorithm [2].

Übeyli, by using SVM reached 99,54% accuracy [21]. Polat and Günes used the LS-SVM (Lest Square SVM) and 98,53% was obtained [15]. Guijarro-Berdias et al. achieved 96,00% classification accuracy rate by applying linear-lest squares [7].

Akay, by using SVM with feauter selection, reached 99,51% classification accuracy rate [1]. Marcano-Cedeño et al., by applying artificial metaplasticity neural network reached 99.26% classification accuracy rate [12]. In this paper, we compare and analyze several kernel functions proposed in the literature by using different values of kernel functions parameters and cost parameters.

This study has been applied to the Wisconsin Breast Cancer Dataset (WBCD) which is a widely studied data set from the field of breast cancer diagnosis. We evaluate the results by calculating the performance measures: classification accuracy rate, sensitivity, specificity, positive and negative predictive values.

In this work we have done two studies: the first one is the realization of a comparison protocol between the different kernels and the second is to reach a high classification accuracy rate in the context of breast cancer diagnosis.

Also, we show that the Cauchy kernel and the Rational Quadratic kernel are identical and give the same results.

Table 2. Kernel functions with parameters

Kernel name	Kernel Parameter
Linear	/
Polynomial	d
Gaussian	σ
Sigmoid	P_1, P_2
Cauchy	σ
Inverse Multiquadric	c
Quadratic	/
Multiquadric	c
Power	d
Rational Quadratic	c
Wave	θ
Spherical	σ

Table 3. WBCD description of attributes

Attribute numbers	Attribute description	Values of attribute
1	Clump thickness	1-10
2	Uniformity of cell size	1-10
3	Uniformity of cell shape	1-10
4	Marginal adhesion	1-10
5	Single epithelial cell size	1-10
6	Bare nuclei	1-10
7	Bland chromatin	1-10
8	Normal nucleoli	1-10
9	Mitoses	1-10

The rest of the paper is organized as follows: First in Section 2, we give an overview of SVM. In Section 3, we recall some kernel functions defined in the literature. In Section 4, we analyze the results of the different kernel function. Finally, we conclude with some perspectives.

2 Overview of Support Vector Machine

Vapnik [4] introduced the Support Vector Machine (SVM) as a learning algorithm aimed at minimizing structural risk. SVM is a method used for data analysis and is employed in both classification and regression tasks.

Given a set of input data, SVM predicts which of two possible classes each input belongs to. To achieve classification, SVM constructs a hyperplane in a high-dimensional space to effectively separate the data into classes.

This hyperplane is positioned to maximize the distance to the nearest training point of any class, ensuring optimal separation. While multiple valid hyperplanes exist, SVM uniquely identifies the optimal hyperplane. The data points that are closest to this maximum margin hyperplane are referred to as Support Vectors.

Identifying this hyperplane involves reformulating the classification problem into a quadratic optimization task, which can be resolved using various algorithms such as Sequential Minimal Optimization, Trust Region, Interior Point, Active-Set, and others. One key benefit of SVM is its effectiveness in high-dimensional spaces, even when the number of dimensions exceeds the number of samples.

2.1 Mathematical Formulation

Given a training data set of N points (x_i, y_i) with input data $x_i \in \mathbb{R}^d$, $i = 1, ..., N$ and output data $y_i \in \{-1, 1\}$ given by an expert. The margin is the distance of closest examples from the line decision (hyperplane). The equation of hyperplane can be written as the set of points X satisfy:

$$\langle w, x_i \rangle + b = 0. \tag{1}$$

The hyperplane that optimally separates the data is the one that minimizes :$\frac{1}{2} w^T w$.

This gives the final standard formulation of an SVM as a minimization problem:

$$\begin{cases} \min \frac{1}{2} w^T w, \\ y_i(\langle w_i, x_i \rangle + b) \geq 1, \ i = 1, ..., N. \end{cases} \tag{2}$$

This represents a quadratic programming optimization challenge. Quadratic optimization problems are a widely recognized category of mathematical optimization problems, with numerous algorithms available for their resolution.

Table 4. The results obtained by different kernel functions with the bestvalue of kernel parameter

Kernel name	Results	Kernel Parameter
Linear	97.40	/
Polynomial	96.54	$d = 3$
Gaussian	99.13	$\sigma = 3$
Sigmoid	96.53	$P_1 = 1, P_2 = -1$
Cauchy	98.27	$\sigma = 10$
Inverse Multiquadric	98.27	$c = 1$
Quadratic	93.59	$c = 5$
Multiquadric	64.93	$c = 9$
Power	34.63	$d = 2$
Rational Quadratic	98.27	$\sigma = 10$
Wave	97.83	$\theta = 2$
Spherical	64.93	$\sigma = 0.1$

The dual problem is obtained by introducing Lagrange musltipliers:

$$\begin{cases} \max \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle, \\ \alpha_i \geq 0, \\ \sum_{i=1}^{N} \alpha_i y_i = 0. \end{cases} \tag{3}$$

Solving equation (3) with constraints equation determines the lagrange multipliers, and the optimal separating hyperplane is given by:

$$w^* = \sum_{i=1}^{N} \alpha_i y_i x_i, \tag{4}$$

$$b^* = -\frac{1}{2} \langle w^*, x_r + x_s \rangle, \tag{5}$$

where x_r and x_s are any support vector from each class satisfying:

$$\alpha_r, \alpha_s > 0, \ y_r = -1, y_s = 1. \tag{6}$$

The hard classifier is then:

$$f(x) = \text{sign}(\langle w^*, x \rangle + b^*). \tag{7}$$

In this study, we choose the Sequential Minimal Optimization algorithm to solve the quadratic problem.

2.2 Sequential Minimal Optimization

The Sequential Minimal Optimization (SMO), introduced by [10, 14] is another widely used algorithm for training Support Vector Machines (SVMs). The basic idea of smo is to decompose the initial problem into sub problems (reducing the working sets to two points).

the optimal solution can be computed analytically for this two points in the working set [5]. Given the current solution $(\alpha_i^{old}, \alpha_j^{old})$, the optimal update is computed to obtain the new solution $(\alpha_i^{new}, \alpha_j^{new})$ by using the following update rule:

$$\alpha_j^{\text{new}} = \alpha_j^{\text{old}} - \frac{y_j(E_i - E_j)}{\eta}, \tag{8}$$

where

$$E_k = f(x_k) - y_k, \tag{9}$$

$$\eta = 2\langle x_i, x_j \rangle - \langle x_i, x_i \rangle - \langle x_j, x_j \rangle, \tag{10}$$

where E_k is the error between the SVM output on the k^{th} example and the true label y_k.

Next we clip α_j^{new} to lie within the range $[L, H]$ i.e $L \leq \alpha_j^{new} \leq H$, to satisfy the constraint that $0 \leq \alpha_j \leq C$:

$$\alpha_j^{\text{new}} = \begin{cases} H & si \ \alpha_j^{\text{new}} \geq H, \\ L & si \ \alpha_j^{\text{new}} \leq L, \\ \alpha_j^{\text{new}} & si \ L < \alpha_j^{\text{new}} < H. \end{cases} \tag{11}$$

The bounds L and H are given by the following:

- If $y_i \neq y_j$, $L = \max(0, \alpha_j^{\text{old}} - \alpha_i^{\text{old}})$, $H = \min(C, C + \alpha_j^{\text{old}} - \alpha_i^{\text{old}})$
- If $y_i = y_j$, $L = \max(0, \alpha_i^{\text{old}} + \alpha_j^{\text{old}} - C)$, $H = \min(C, \alpha_i^{\text{old}} + \alpha_j^{\text{old}})$.

Finally, having solved for α_j^{new}, the value of α_i^{new} is given by:

$$\alpha_i^{\text{new}} = \alpha_i^{\text{old}} + y_i y_j (\alpha_i^{\text{old}} - \alpha_j^{\text{old}}). \tag{12}$$

The algorithm proceeds as follows:

1. Find a Lagrange multiplier α_1 that violates the $Karush - Kuhn - Tucker \ (KKT)$ conditions for the optimization problem.

Table 5. The results obtained by different kernel functions with the bestvalue of SVM model parameter

Kernel name	cost parameter C
Linear	1
Polynomial	1
Gaussian	13
Sigmoid	1
Cauchy	6
Inverse Multiquadric	2
Quadratic	5
Multiquadric	1
Power	2
Rational Quadratic	6
Wave	1
Spherical	1

Table 6. Sensitivity and specificity calculated for the four best kernel functions

Kernel name	Sensitivity	Specificity
Gaussian	98.67	100
Inverse M.	98.67	97.53
Cauchy	98.67	97.53
Wave	98.67	96.30

2. Pick a second multiplier α_2 and optimize the pair (α_1, α_2).
3. Repeat steps 1 and 2 until convergence.

Upon satisfaction of the Karush-Kuhn-Tucker (KKT) conditions by all Lagrange multipliers within a specified user-defined tolerance, the problem is considered solved.

While this algorithm ensures convergence, heuristics are employed to select the pair of multipliers to expedite convergence. To achieve optimal performance, certain parameters in SVM must be meticulously chosen. These parameters include:

– The regularization parameter C, which controls the trade-off between errors of the SVM on training data and margin maximization [20].
– The parameters of the kernel functions.
– The choice of the kernel affect the performance.

3 Kernel Functions

However, in 1992, V. Vapnik et al. proposed a method to generate nonlinear classifiers by employing the kernel trick with maximum-margin hyperplanes.

Kernel functions enable a nonlinear transformation of data into a linear separation of examples in a new space known as the ”feature space,” which is high-dimensional.

This characteristic enhances the likelihood of discovering a separating hyperplane. Nevertheless, in this new space, the goal is to find the following hyperplane:

$$h(x) = \langle w, \Phi(x) \rangle + b. \tag{13}$$

We arrive at the following optimization problem:

$$\begin{cases} \max \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \Phi(x_i), \Phi(x_j) \rangle, \\ \alpha_i \geq 0, \\ \sum_{i=1}^{N} \alpha_i y_i = 0. \end{cases} \tag{14}$$

By introducing the notion of kernel function we have:

$$k(x_i, x_j) = \Phi(x_i)\Phi(x_j). \tag{15}$$

The expression of the hyperplane will be defined as follows:

$$f(x) = \sum_i \alpha_i y_i \Phi(xi)\Phi(xj) + b. \tag{16}$$

Under these conditions we didn't need to know the transformation Φ and the calculation will be much less expensive.

We can directly construct a kernel function by respecting certain condition defined by the theorem of Mercer which states that: a kernel $k(x_i, x_j)$ is a symmetric continuous function that maps two variables to a real value and k must be positive semi definite.

Unfortunately, this theoretical condition is difficult to verify, it does not provide guidance for the construction of kernels or on the transformation

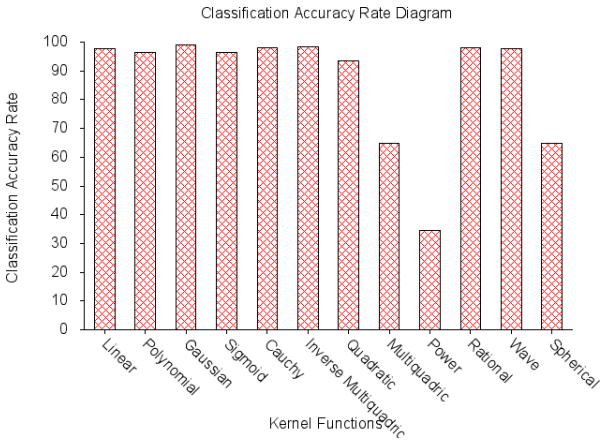


Fig. 2. The figure shows the result obtained by the different kernel

Φ. Several researches has been devoted to constructing a kernel more exotic and adapted to a special problem. In table 2, we show the several kernel functions proposed in the literature:

4 Experimentation

In this research, we analyze and evaluate the performance of different kernel functions in the SVM algorithm by using different values of kernel functions parameters and SVM model parameters. We use the SMO algorithm to solve the quadratic problem of maximizing the margin.

This analysis is conducted using the publicly accessible breast cancer database known as WBCD (Wisconsin Breast Cancer Database), which originates from the work undertaken at the University of Wisconsin Hospital.

This set of data was taken from Fine Needle Aspirates (FNA) of humain breast tissue classified as benign and maligne. The WBCD database contains 699 clinical cases, there is 458 (65,50%) benign cases and 241 (34,50%) malignant cases.

The data of WBCD contains 16 instances with missing attirubte values which led us to limir our experimentation to 683 clinical cases. Nevertheless, the class has a distribution of 444 (65%) besnigs cases and 239 (35%) malignant cases. Each instance in the database has a nine attributes; each attributes has an integer value between 1 and 10.

Table 7. Positive predictive value and negative predictive value calculated for the four best kernel functions

Kernel name	Pos. Pre. Val.	Neg. Pre. Val.
Gaussian	100	97.59
Inverse M.	98.67	97.53
Cauchy	98.67	97.53
Wave	98.01	97.50

In the following table, we detailed the attributes of WBCD: The SVM method consists of two phases: training and testing. To randomly divide the database into two parts, we employ the holdout method, a form of cross-validation.

This method randomly partitions the initial data into two sets: the training set and the testing set. Less than one-third of the initial data is allocated for testing purposes. With the holdout method, we obtained 455 samples(65,10%) for the training phase and 244 samples (34,90%) for the testing phase.

In the first step, we analyze and compare the results obtained by the different kernel functions in terms of accuracy classification rate. The values of kernel functions parameters were chosen by experimentation.

For the cost parameter C (regularization parameters that control the flexibility), we have varied its value between 0.01 and 1000 then we record the values which give a good accuracy classification rate.

In the second step, we take the four kernel functions which have given a good results and we analyze these kernels in the function of: sensitivity, specificity, positive and negative predictive values. The parameters mesearse are calculated using the following equations:

where:

$$\begin{aligned} NTP: &\quad Number\ of\ True\ Positives \\ NTN: &\quad Number\ of\ True\ Negatives \\ NFP: &\quad Number\ of\ False\ Positives \\ NFN: &\quad Number\ of\ False\ Negatives \end{aligned}$$

Table 8. Classification accuracy rate (CAR) obtained with SVM by using the Gaussian kernel and other classifiers

Author and years	CAR
Quinlan (1996)	94.74
Hamiton et al. (1996)	95.00
Nauck and Kruse (1999)	95.06
Albrecht et al. (2002)	98.80
Polat and Günes (2007)	98.53
Guijarro-Berdias et al. (2007)	96.00
Akay (2009)	99.51
A. Marcano-Cedeño (2011)	99.26
This Study	99.13

Classification accuracy: $\frac{NTP+NTN}{NTP+NTN+NFP+NFN}$

Sensitivity: $\frac{NTP}{NTP+NFN}$

Specificity: $\frac{NTN}{NFP+NTN}$

Positive Predictive Value: $\frac{NTP}{NTP+NFP}$

Negative Predictive Value: $\frac{NTN}{NTN+NFN}$

In table 5, we present the good results (the high value of: classification accuracy rate and its kernel parameters) obtained by each kernel function after several trials and evaluation of different values of the kernel functions parameters with SVM model parameters. We clearly observe, that the Gaussian, Cauchy, inverse multiquadratic, rational quadratic, linear and polynomial kernels have given a good results with an advantage for Gaussian kernel function (99,13%).

We show also, that the Cauchy, inverse multiquadratic and rational quadratic kernel have nearly the same classification accuracy rate. The low classification accuracy rate is registered for the multiquadratic and spherical kernel function with 64,93%.

During the assessments, we show that the results obtained by the rational quadratic kernel and the Cauchy kernel are identical despite the different tests with different training and testing set. Therefore, we will show that the rational quadratic kernel and the Cauchy kernel are identical and converge to the same value:

$$\begin{aligned} k(x,y) &= \frac{1}{1+\frac{\|x-y\|^2}{c}}, \\ &= \frac{1}{\frac{c+\|x-y\|^2}{c}}, \\ &= \frac{c}{c+\|x-y\|^2}. \end{aligned} \tag{17}$$

The Rational Quadratic kernel is defined as:

$$\begin{aligned} k(x,y) &= 1-\frac{\|x-y\|^2}{\|x-y\|^2+c}, \\ &= \frac{\|x-y\|^2+c}{\|x-y\|^2+c}-\frac{\|x-y\|^2}{\|x-y\|^2+c}, \\ &\text{with } 1=\frac{\|x-y\|^2+c}{\|x-y\|^2+c}, \\ &= \frac{c}{\|x-y\|^2+c}. \end{aligned} \tag{18}$$

Finally, we can say that using the rational quadratic kernel function gives the same results as the Cauchy kernel function. So, the transformation of data by these kernel are the same and give the identical new space. In table 7, we show: sensitivity, specificity, positive and negative predictive values obtained by the four kernels which have given a high diagnostic accuracy.

Table 8, gives the classification accuracies of SVM with Gaussian kernel and previous methods applied to the same database.

5 Conclusion

In this study, we have analyze and compared the performance of several kernel on the support vector machine in the context of breast cancer diagnosis. We conducted our experimentation on the WBCD database. We have shown that the Gaussian kernel has given a good results in term of classification accuracy rate (99,13%). The Gaussian kernel have a great performance with: 98,67% in sensitivity and 100% in specificity. Also, we have shown that the both Cauchy kernel and the Rational Quadratic kernel are the same and give the same results.

References

1. **Akay, M. F. (2009).** Support vector machines combined with feature selection for breast cancer diagnosis. Expert Systems with Applications, Vol. 36, No. 2, pp. 3240–3247. DOI: 10.1016/j.eswa.2008.01.009.
2. **Albrecht, A. A., Lappas, G., Vinterbo, S. A., Wong, C. K., Ohno-Machado, L. (2002).** Two applications of the lsa machine. Proceedings of the 9th international conference on neural information processing, pp. 184–189. DOI: 10.1109/ICONIP.2002.1202156.
3. **Brook, A., El-Yaniv, R., Isler, E., Kimmel, R., Meir, R., Peleg, D. (2006).** Breast cancer diagnosis from biopsy images using generic features and SVMs. IEEE Transactions on Information Technology in Biomedicine.
4. **Cortes, C., Vapnik, V. (1995).** Support-vector networks. Machine Learning, Vol. 20, pp. 273–297. DOI: 10.1007/Bf00994018.
5. **Flake, G. W., Lawrence, S. (2002).** Efficient SVM regression training with SMO. Machine Learning, Vol. 46, pp. 271–290. DOI: 10.1023/A:1012474916001.
6. **Forouzanfar, M. H., Foreman, K. J., Delossantos, A. M., Lozano, R., Lopez, A. D., Murray, C. J., Naghavi, M. (2011).** Breast and cervical cancer in 187 countries between 1980 and 2010: A systematic analysis. The Lancet, Vol. 378, No. 9801.
7. **Guijarro-Berdias, B., Fontenla-Romero, O., Perez-Sanchez, B., Fraguela, P. (2007).** A linear learning method for multilayer perceptrons using leastsquares. Lecture Notes in Computer Science, pp. 365–374. DOI: 10.1007/978-3-540-77226-2_38.
8. **Guyon, I., Weston, J., Barnhill, S., Vapnik, V. (2002).** Gene selection for cancer classification using support vector machines. Machine Learning, Vol. 46, pp. 389–422. DOI: 10.1023/A:1012487302797.
9. **Hamilton, H. J., Shan, N., Cercone, N. (1996).** A rule induction algorithm based on approximate classification. Journal of Computer and System Science, Vol. 20, No. 11, pp. 34–50.
10. **Keerthi, S. S., Shevade, S. K., Bhattacharyya, C., Murthy, K. R. K. (2001).** Improvements to platt's SMO algorithm for SVM classifier design. Neural Computation, Vol. 13, No. 3. DOI: 10.1162/089976601300014493.
11. **Mallika, R., Saravanan, V. (2010).** An SVM based classification method for cancer data using minimum microarray gene expressions. International Journal of Computer and Information Engineering, Vol. 4, No. 2, pp. 266–270.
12. **Marcano-Cedeno, A., Quintanilla-Domínguez, J., Andina, D. (2011).** WBCD breast cancer database classification applying artificial metaplasticity neural network. Expert Systems with Applications, Vol. 38, No. 8, pp. 9573–9579. DOI: 10.1016/j.eswa.2011.01.167.
13. **Nauck, D., Kruse, R. (1999).** Obtaining interpretable fuzzy classification rules from medical data. Artificial Intelligence in Medicine, Vol. 16, No. 2. DOI: 10.1016/S0933-3657(98)00070-0.
14. **Platt, J. C. (1999).** Fast training of support vector machines using sequential minimal optimization. Advances in Kernel Methods - Support Vector Learning. DOI: 10.7551/mitpress/1130.003.0016.
15. **Polat, K., Günes, S. (2007).** Breast cancer diagnosis using least square support vector machine. Digital Signal Processing, Vol. 17, No. 4, pp. 694–701. DOI: 10.1016/j.dsp.2006.10.008.
16. **Quinlan, J. R. (1996).** Improved use of continuous attributes in C4.5. Journal of Artificial Intelligence Research, Vol. 4, pp. 77–90. DOI: 10.1613/jair.279.
17. **Rejani, Y. I. A., Selvi, S. T. (2009).** Early detection of breast cancer using SVM classifier technique. International Journal on Computer

Science and Engineering, Vol. 1, No. 3. DOI: 10.48550/arXiv.0912.2314.

18. **Rychetsky, M. (2001).** Algorithms and architectures for machine learning based on regularized neural networks and support vector approaches. Shaker, Germany.
19. **Shah, S., Kusiak, A. (2002).** Cancer gene search with data-mining and genetic algorithms. Computers in Biology and Medicine, Vol. 37, No. 2, pp. 251–261. DOI: 10.1016/j.compbiomed.2006.01.007.
20. **Shawe-Taylor, J., Cristianini, N. (2004).** Kernel Methods for Pattern Analysis. Cambridge University Press.
21. **Übeyli, E. D. (2007).** Implementing automated diagnostic systems for breast cancer detection. Expert Systems with Applications, Vol. 33, No. 4, pp. 1054–1062. DOI: 10.1016/j.eswa.2006.08.005.

Article received on 19/12/2023; accepted on 18/04/2024.
* *Corresponding author is Seyyid Ahmed-Medjahed.*

ISSN 2007-9737

Deflexiones en trabes de sección transversal rectangular con cartelas parabólicas sometidas a una carga uniformemente distribuida

Gilberto Ramírez-Muñoz, Arnulfo Luévanos-Rojas*,
Sandra López-Chavarría, Manuel Medina-Elizondo

Universidad Autónoma de Coahuila,
Instituto de Investigaciones Multidisciplinaria,
México

gyram9@gmail.com, {arnulfol_2007, sandylopez5}@hotmail.com,
drmanuelmedina@yahoo.com.mx

Resumen. Este trabajo muestra las ecuaciones para una viga de sección rectangular con cartelas parabólicas sometida a un momento en cada apoyo y una carga uniformemente distribuida tomando en cuenta las deformaciones por flexión y cortante para obtener las rotaciones y los desplazamientos perpendiculares al eje longitudinal en cualquier punto de la viga, que es el principal aporte de esta investigación. El modelo actual considera solo las deformaciones por flexión para obtener las rotaciones y los desplazamientos perpendiculares al eje longitudinal de la viga. La metodología se desarrolla bajo el concepto de que la integral del momento es la pendiente o rotación y la segunda integración del momento es el desplazamiento vertical de la viga. Asimismo, se hace una comparación entre el modelo propuesto y el modelo actual para vigas simplemente apoyadas y vigas empotradas en ambos extremos con respecto al desplazamiento vertical máximo y su ubicación en la viga. Los resultados muestran que el modelo propuesto es mayor para los dos tipos de vigas para el desplazamiento vertical máximo con respecto al modelo actual. Por tanto, el modelo propuesto es más seguro y más adecuado con respecto al modelo actual para el análisis estructural, ya que las fuerzas cortantes y los momentos flectores están presentes en todas las estructuras y se producen deformaciones por flexión y cortante.

Palabras clave. Trabes rectangulares, cartelas parabólicas, deformaciones por flexión y cortante, rotaciones, desplazamientos verticales.

Deflections in Beams of Rectangular Cross Section with Parabolic Haunches Subjected to a Uniformly Distributed Load

Abstract. This work shows the equations for a rectangular section beam with parabolic haunches subjected to a moment at each support and a uniformly distributed load taking into account the shear and bending deformations to obtain the rotations and the perpendicular displacements to the longitudinal axis at any point of the beam, which is the main contribution of this research. The current model considers only the bending deformations to obtain the rotations and the perpendicular displacements to the longitudinal axis of the beam. The methodology is developed under the concept that the integral of the moment is the slope or rotation and the second integration of the moment is the vertical displacement of the beam. Also, a comparison is made between the proposed model and the current model for beams fixed at both ends with respect to the maximum vertical displacement and its location on the beam. Results show that the proposed model is greater for beams fixed at both ends in the maximum vertical displacement with respect to the current model. Therefore, the proposed model is safer and more suitable with respect to the current model for structural analysis, since shear forces and bending moments are present in all structures and deformations due to bending and shear occur.

Keywords. Rectangular beams, parabolic haunches, bending and shear deformations, rotations, vertical displacements.

1. Introducción

La deformación de vigas de concreto reforzado y vigas de acero estructural es una medida importante de su rendimiento de servicio, ya que se requiere específicamente cumplir con los códigos de diseño actuales. Generalmente, las deformaciones de las vigas consisten en deformaciones por flexión y cortante.

En ingeniería estructural, existen dos criterios de diseño para las vigas: la resistencia y la capacidad de servicio.

Las principales publicaciones sobre vigas en voladizo no prismáticas son las de Lee [1], Dado y Al-Sadder [2], Borboni y De Santis [3], Banerjee *et al*. [4], Solano-Carrillo [5], Chen [6], Yau [7], Brojan *et al*. [8].

Algunos investigadores han estudiado el problema de la curva elástica para vigas de secciones prismáticas rectangulares simplemente apoyadas sometidas a una carga uniformemente distribuida y/o carga concentrada tomando en cuenta las deformaciones por flexión y cortante para obtener las deflexiones en cualquier lugar de la viga (rotaciones y desplazamientos perpendiculares al eje longitudinal) [9, 10].

Varios autores han desarrollado modelos matemáticos y ayudas de diseño para vigas rectangulares con cartelas rectas o cartelas parabólicas en los extremos para las vigas sometidas a una carga uniformemente distribuida o carga concentrada tomando en cuenta las deformaciones a flexión y cortante [11-18]. Estos documentos muestran los factores para los momentos de empotramiento, factores de arrastre o factores de transporte y factores de rigidez.

Otros autores han presentado modelos óptimos para obtener las dimensiones y el acero de refuerzo longitudinal para vigas rectangulares con cartelas rectas o cartelas parabólicas bajo carga uniformemente distribuida o carga concentrada y momentos en los extremos [19, 20]. Las revisiones de la literatura de las investigaciones desarrolladas y/o comparadas por software se muestran a continuación:

Majumder y Kumar [21] analizaron la deflexión máxima de una viga simplemente apoyada bajo diferentes tipos de carga.

Las cargas son: a) Carga concentrada en el centro de la viga; b) Carga uniformemente distribuida; c) Carga triangularmente distribuida. El análisis teórico se realizó mediante la Teoría de Euler-Bernoulli (solo considera las deformaciones por flexión) y se comparó con el software ANSYS 14.0. Al comparar los resultados numéricos con los obtenidos por el software ANSYS 14.0, se ha demostrado una excelente precisión del presente método. Si los resultados fueron más precisos entre la teoría de Euler-Bernoulli y el software ANSYS 14.0 como se muestra en las conclusiones, entonces las deformaciones por cortante no se consideran en el software ANSYS 14.0.

Debnath y Debnath [22] estudiaron la deflexión máxima para diferentes vigas de sección transversal rectangular uniforme, y los tipos de vigas son: a) Viga simplemente apoyada con una carga uniformemente distribuida; b) Viga simplemente apoyada con una carga concentrada en el centro; c) Viga en voladizo con carga uniformemente distribuida; d) Viga en voladizo con carga concentrada en el extremo.

El cálculo teórico se realizó según la Teoría de Euler-Bernoulli (solo considera las deformaciones por flexión) y el análisis computacional se realizó con el software ANSYS 14.0. Los datos considerados para todas las vigas son: L = 100 m, b = 10 m, h = 10 m, v = 0.3, E = 2×107 N/m^2, F = 500 N. Los elementos sólidos estudiados fueron 188, 189, 185 y 285.

El resultado más preciso fue medido por el elemento sólido 189 seguido por el elemento sólido 188 y otros elementos sólidos. Si los resultados fueron precisos entre la teoría de Euler-Bernoulli y el software ANSYS 14.0, entonces las deformaciones por cortante se ignoran en el software ANSYS 14.0.

Sihua *et al*. [23] presentaron el análisis no lineal para una viga de concreto reforzado con ayuda del software de análisis de elementos finitos ABAQUS. La viga es simplemente apoyada y tiene 1500 mm de largo; con una sección de 180×100 mm, la resistencia del concreto es C25, el refuerzo longitudinal y estribos adoptados HPB235 reforzados.

En este análisis de vigas simplemente apoyadas, se ha introducido a fondo el modelo de plasticidad del daño del concreto en ABAQUS.

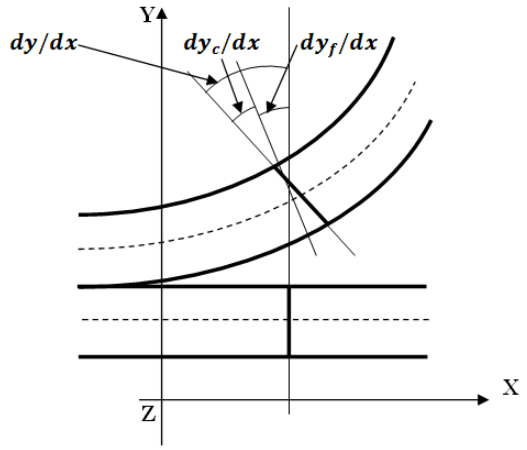


Fig. 1. Deformación de un elemento de viga

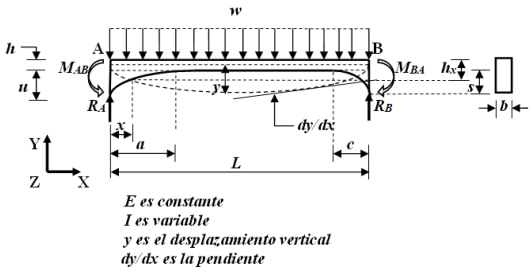


Fig. 2. Viga sometida a un momento en cada extremo y una carga uniformemente distribuida

Finalmente, los resultados de la experimentación y el análisis ABAQUS se compararon en un diagrama, y la carga alcanzó la capacidad de 24 kN, el valor de deflexión en el centro de la longitud de la viga es de 10.521 mm de ABAQUS y 12.795 mm de prueba.

Si se hubieran considerado las deformaciones de cortante en el software ABAQUS, los resultados del software ABAQUS estarían más cerca de la prueba experimental, porque estas deformaciones tienden a aumentar a las deformaciones totales para vigas.

En este trabajo se presenta un modelo matemático de la curva elástica para vigas rectangulares con cartelas parabólicas bajo carga uniformemente distribuida y momentos en los extremos considerando las deformaciones por flexión y cortante (Teoría de Timoshenko) para obtener las rotaciones y desplazamientos en la viga, que es la parte principal de esta investigación. El modelo propuesto se muestra en tres partes para la viga de $0 \le x \le a$, $a \le x \le L - c$ y $L - c \le x \le L$. El modelo actual de la curva elástica para vigas rectangulares bajo carga uniformemente distribuida considera solo las deformaciones por flexión (Teoría de Euler- Bernoulli).

2. Modelo propuesto

La Figura 1 muestra la diferencia entre la teoría de Timoshenko y la teoría de Euler-Bernoulli. La primera teoría incluye el efecto de los esfuerzos de flexión y cortante sobre la deformación ($dy/dx = dy_c/dx + dy_f/dx$), y la segunda teoría incluye el efecto de los esfuerzos de flexión en la deformación ($dy/dx = dy_f/dx$) [24, 25].

La teoría de Timoshenko considera las deformaciones por flexión y cortante, esto es válido para los miembros cortos y largos. La ecuación de la curva elástica se presenta de la siguiente manera [24, 25]:

$$\frac{d^2y}{dx^2} = \frac{d^2y_c}{dx^2} + \frac{d^2y_f}{dx^2}, \tag{1}$$

$$\frac{d^2y}{dx^2} = -\frac{w}{GA_{cx}} - \frac{M_z}{EI_z}, \tag{2}$$

donde: G es el módulo de cortante, y es el desplazamiento total, y_c es el desplazamiento por cortante debido a la fuerza cortante, y_f es el desplazamiento por flexión debido al momento, A_{cx} es el área de cortante, E es el módulo de elasticidad, M_z es el momento de flexión alrededor del eje Z, I_z es el momento de inercia alrededor del eje Z.

Las rotaciones en cualquier parte de la viga por integración de la ecuación (2) se obtienen:

$$\frac{dy}{dx} = -\int \frac{w}{GA_{cx}}dx - \int \frac{M_z}{EI_z}dx. \tag{3}$$

En la Figura 2 se muestra la viga AB sometida a un momento en cada extremo y una carga uniformemente distribuida, así como su sección transversal rectangular teniendo en cuenta que el ancho “b” es constante y la altura “h_x” es variable de forma parabólica en sus extremos y constante en la parte central.

Tabla 1. Propiedades de la sección rectangular con cartelas parabólicas

Concepto	**Ecuaciones**		
Intervalo	***0 ≤ x ≤ a***	***a ≤ x ≤ L – c***	***L – c ≤ x ≤ L***
$h_x(x)$	$\frac{a^2h+u(x-a)^2}{a^2}$	h	$\frac{c^2h+s(x-L+c)^2}{c^2}$
$A_{cx}(x)$	$\frac{5b[a^2h+u(x-a)^2]}{6a^2}$	$\frac{5bh}{6}$	$\frac{5b[c^2h+s(x-L+c)^2]}{6c^2}$
$I_z(x)$	$\frac{b[a^2h+u(x-a)^2]^3}{12a^6}$	$\frac{bh^3}{12}$	$\frac{b[c^2h+s(x-L+c)^2]^3}{12c^6}$

Los valores de las reacciones en los apoyos R_A y R_B se obtienen de las siguientes ecuaciones:

$$R_A=\frac{M_{AB}-M_{BA}}{L}+\frac{wL}{2}, \tag{4}$$

$$R_B=\frac{wL}{2}-\frac{M_{AB}-M_{BA}}{L}. \tag{5}$$

El momento alrededor del eje Z a una distancia x es:

$$M_z=R_Ax-\frac{wx^2}{2}-M_{AB}. \tag{6}$$

Sustituyendo la ecuación (6) en la ecuación. (3) se obtiene:

$$\frac{dy}{dx}=-\int\frac{w}{GA_{cx}}dx-\int\frac{R_Ax-\frac{wx^2}{2}-M_{AB}}{EI_z}dx. \tag{7}$$

La Tabla 1 muestra las ecuaciones de las alturas "$h_x(x)$" a una distancia "x", áreas de cortante "$A_{cx}(x)$" a una distancia "x", y el momento de inercia "$I_z(x)$" alrededor del eje "z" a una distancia "x" para cada intervalo.

Sustituyendo las propiedades de la Tabla 1 en la ecuación (7) para cada intervalo, y resolviendo la integral se obtienen las rotaciones en cualquier lugar (dy/dx), y posteriormente se desarrolla la integral para obtener los desplazamientos en cualquier lugar de la viga (y).

Las ecuaciones para el primer intervalo de $0 \le x \le a$ se muestra en las ecuaciones (8) y (9). Las ecuaciones para el segundo intervalo de $a \le x \le L-c$ se muestra en las ecuaciones (10) y (11). Las ecuaciones para el tercer intervalo de $L-c \le x \le L$ se muestra en las ecuaciones (12) y (13):

$$\frac{dy}{dx}=\frac{12}{Eb}\left\{\left[\frac{3au(2R_Aa-wa^2-2M_{AB})-wa^3h}{16h^2u\sqrt{hu}}\right]Atan\left[\frac{u(a-x)}{a\sqrt{hu}}\right]-\frac{wa^4[a^2h(3a+x)+u(a-x)^3]-4R_Aa^6h}{16hu[a^2h+u(a-x)^2]^2}+\frac{a^2(2R_Aa-wa^2-2M_{AB})[3u(a-x)^3+5a^2h(a-x)]}{16h^2[a^2h+u(a-x)^2]^2}\right\}+\frac{6w}{5Gb}\left\{\frac{a}{\sqrt{hu}}Atan\left[\frac{u(a-x)}{a\sqrt{hu}}\right]\right\}+C_1, \tag{8}$$

$$y=\frac{12}{Eb}\left\{\left[\frac{a^3h[w(3a-x)-2R_A]-3au(2R_Aa-wa^2-2M_{AB})(a-x)}{16h^2u\sqrt{hu}}\right]Atan\left[\frac{u(a-x)}{a\sqrt{hu}}\right]+\left[\frac{wa^4h-3a^2u(2R_Aa-wa^2-2M_{AB})}{32h^2u^2}\right]ln(h^2u)+\frac{wa^6(h+u)+2a^4u(R_Ax-wax-M_{AB})}{16hu^2[a^2h+u(a-x)^2]}\right\}+\frac{6w}{5Gb}\left\{\frac{a^2}{2u}ln\left[\frac{a^2h+u(a-x)^2}{a^2h}\right]-\frac{a(a-x)}{\sqrt{hu}}Atan\left[\frac{u(a-x)}{a\sqrt{hu}}\right]\right\}+C_1x+C_2, \tag{9}$$

$$\frac{dy}{dx}=\frac{12}{Eb}\left(\frac{wx^3}{6h^3}-\frac{R_Ax^2}{2h^3}+\frac{M_{AB}x}{h^3}\right)+\frac{6w}{5Gb}\left(-\frac{x}{h}\right)+C_3, \tag{10}$$

$$y=\frac{12}{Eb}\left(\frac{wx^4}{24h^3}-\frac{R_Ax^3}{6h^3}+\frac{M_{AB}x^2}{2h^3}\right)+\frac{6w}{5Gb}\left(-\frac{x^2}{2h}\right)+C_3x+C_4, \tag{11}$$

$$\frac{dy}{dx}=\frac{12}{Eb}\left\{\left[\frac{3cs[2R_A(L-c)-w(L-c)^2-2M_{AB}]-wc^3h}{16h^2s\sqrt{hs}}\right]Atan\left[\frac{s(L-x-c)}{c\sqrt{hs}}\right]+\frac{c^2[2R_A(L-c)-w(L-c)^2-2M_{AB}][5c^2h(L-x-c)+3s(L-x-c)^3]}{16h^2[c^2h+s(L-x-c)^2]^2}-\frac{wc^4\{c^2h[3(L-c)+x]+s(L-x-c)^3\}-4R_Ac^6h}{16hs[c^2h+s(L-x-c)^2]^2}\right\}+\frac{6w}{5Gb}\left\{\frac{c}{\sqrt{hs}}Atan\left[\frac{s(L-x-c)}{c\sqrt{hs}}\right]\right\}+C_5, \tag{12}$$

$$y=\frac{12}{Eb}\left\{\left[\frac{c^3h\{w[3(L-c)-x]-2R_A\}-3cs[2R_A(L-c)-w(L-c)^2-2M_{AB}](L-x-c)}{16h^2s\sqrt{hs}}\right]At+\left[\frac{wc^4h-3c^2s[2R_A(L-c)-w(L-c)^2-2M_{AB}]}{32h^2s^2}\right]ln(ch^2s)+\frac{wc^4[c^2h+s(L-c)^2]+2c^4s[R_Ax-w(L-c)x-M_{AB}]}{16hs^2[c^2h+s(x-L+c)^2]}\right\}+\frac{6w}{5Gb}\left\{\frac{c^2}{2s}ln\left[\frac{c^2h+s(L-x-c)^2}{c^2h}\right]-\frac{c(L-x-c)}{\sqrt{hs}}Atan\left[\frac{s(L-x-c)}{c\sqrt{hs}}\right]\right\}+C_5x+C_6. \tag{13}$$

Las seis condiciones conocidas que debe cumplir la viga para obtener las constantes de integración son:

1) Sustituyendo la condición x = 0 e y = 0 en la ecuación (9).

2) Sustituyendo la condición x = a en las ecuaciones (8) y (10), estas dos ecuaciones se igualan, porque las rotaciones "dy/dx" en este punto deben ser las mismas.

3) Sustituyendo la condición x = a en las ecuaciones (9) y (11), estas dos ecuaciones se igualan, porque los desplazamientos "y" en este punto deben ser los mismos.

4) Sustituyendo la condición x = L – c en las ecuaciones (10) y (12), estas dos ecuaciones se igualan, porque las rotaciones "dy/dx" en este punto deben ser las mismas.

5) Sustituyendo la condición x = L – c en las ecuaciones (11) y (13), estas dos ecuaciones se igualan, porque los desplazamientos "y" en este punto deben ser los mismos.

6) Sustituyendo la condición x = L e y = 0 en la ecuación (13).

Ahora, sustituyendo las seis condiciones conocidas en las ecuaciones correspondientes, se generan seis ecuaciones para obtener las constantes de integración.

Las constantes de integración se muestran a continuación:

$$C_2 = \frac{12}{Eb}\left\{\left[\frac{a^3h(2R_A-3wa)+3a^2u(2R_Aa-wa^2-2M_{AB})}{16h^2u\sqrt{hu}}\right]Atan\left(\sqrt{\frac{u}{h}}\right) - \frac{wa^4(h+u)-2M_{AB}a^2u}{16huh^2(h+u)} - \left[\frac{wa^4h-3a^2u(2R_Aa-wa^2-2M_{AB})}{32h^2u^2}\right]ln(h^2u)\right\} + \frac{6w}{5Gb}\left\{\frac{a^2}{\sqrt{hu}}Atan\left(\sqrt{\frac{u}{h}}\right)-\frac{a^2}{2u}ln\left(\frac{h+u}{h}\right)\right\}, \quad (14)$$

$$C_4 = \frac{12}{Eb}\left\{\left[\frac{a^3h(2R_A-3wa)+3a^2u(2R_Aa-wa^2-2M_{AB})}{16h^2u\sqrt{hu}}\right]Atan\left(\sqrt{\frac{u}{h}}\right) - \frac{R_Aa^3(3h+8u)}{24h^3u}+\frac{wa^4(3h+2u)}{16h^3u} + \frac{M_{AB}a^2(3h+4u)}{8h^3(h+u)}\right\} + \frac{6w}{5Gb}\left\{\frac{a^2}{\sqrt{hu}}Atan\left(\sqrt{\frac{u}{h}}\right)-\frac{a^2}{2h}-\frac{a^2}{2u}ln\left(\frac{h+u}{h}\right)\right\}, \quad (15)$$

$$\begin{aligned}C_6 &= \frac{12}{Eb}\left\{\frac{R_A[8s(L-c)^3+3c^2h(L-c)]}{24h^3s}-\frac{w[2s^2(L-c)^4+3c^2hs(L-c)^2+c^4h^2]}{16h^3s^2}\right.\\ &-\frac{M_{AB}[4s(L-c)^2-c^2h]}{8h^3s}-\frac{R_Aa^3(3h+8u)}{24h^3u}+\frac{wa^4(3h+2u)}{16h^3u}\\ &-\left[\frac{wc^4h-3c^2s[2R_A(L-c)-w(L-c)^2-2M_{AB}]}{32h^2s^2}\right]ln(ch^2s)+\frac{M_{AB}a^2(3h+4u)}{8h^3(h+u)}\\ &-\left[\frac{a^3h(3wa-2R_A)-3a^2u(2R_Aa-wa^2-2M_{AB})}{16h^2u\sqrt{hu}}\right]Atan\left(\sqrt{\frac{u}{h}}\right)\Big\}\\ &+\frac{6w}{5Gb}\left\{\frac{(L-c)^2-a^2}{2h}-\frac{a^2}{2u}ln\left(\frac{h+u}{h}\right)+\frac{a^2}{\sqrt{hu}}Atan\left(\sqrt{\frac{u}{h}}\right)\right\},\end{aligned} \quad (16)$$

$$\begin{aligned}C_5 &= \frac{12}{Eb}\left\{\frac{w\{(h+s)(L-c)^2[2s(L-c)^2+3c^2h]+L^2c^2h^2\}}{16Lh^3s(h+s)}\right.\\ &-\frac{R_A\{(h+s)(L-c)[8s(L-c)^2+3c^2h]+3Lc^2h^2\}}{24Lh^3s(h+s)}-\frac{wa^4(3h+2u)}{16Lh^3u}+\frac{R_Aa^3(3h+8u)}{24Lh^3u}\\ &-\frac{M_{AB}a^2(3h+4u)}{8Lh^3(h+u)}+\frac{M_{AB}\{4(h+s)(L-c)^2-c^2h\}}{8Lh^3(h+s)}\\ &+\left[\frac{a^3h(3wa-2R_A)-3a^2u(2R_Aa-wa^2-2M_{AB})}{16Lh^2u\sqrt{hu}}\right]Atan\left(\sqrt{\frac{u}{h}}\right)\\ &+\left[\frac{c^3h\{w(2L-3c)-2R_A\}+3c^2s[2R_A(L-c)-w(L-c)^2-2M_{AB}]}{16Lh^2s\sqrt{hs}}\right]Atan\left(\sqrt{\frac{s}{h}}\right)\Big\}\\ &+\frac{6w}{5Gb}\left\{\frac{a^2-(L-c)^2}{2Lh}+\frac{a^2}{2Lu}ln\left(\frac{h+u}{h}\right)-\frac{c^2}{2Ls}ln\left(\frac{h+s}{h}\right)-\frac{a^2}{L\sqrt{hu}}Atan\left(\sqrt{\frac{u}{h}}\right)\right.\\ &+\left.\frac{c^2}{L\sqrt{hs}}Atan\left(\sqrt{\frac{s}{h}}\right)\right\},\end{aligned} \quad (17)$$

$$\begin{aligned}C_3 &= \frac{12}{Eb}\left\{\frac{R_Aa^3(3h+8u)}{24Lh^3u}+\frac{R_A\{(h+s)[4s(L-c)^2(L+2c)+3c^2h(L+c)]-3Lc^2h^2\}}{24Lh^3s(h+s)}\right.\\ &-\frac{M_{AB}a^2(3h+4u)}{8Lh^3(h+u)}-\frac{wa^4(3h+2u)}{16Lh^3u}\\ &-\frac{w\{(h+s)(L-c)[2s(L-c)^2(L+3c)+3c^2h(L+3c)]-3L^2c^2h^2\}}{48Lh^3s(h+s)}\\ &-\frac{M_{AB}[4(h+s)(L^2-c^2)+c^2h]}{8Lh^3(h+s)}\\ &+\left[\frac{a^3h(3wa-2R_A)-3a^2u(2R_Aa-wa^2-2M_{AB})}{16Lh^2u\sqrt{hu}}\right]Atan\left(\sqrt{\frac{u}{h}}\right)\\ &+\left[\frac{c^3h\{w(2L-3c)-2R_A\}+3c^2s[2R_A(L-c)-w(L-c)^2-2M_{AB}]}{16Lh^2s\sqrt{hs}}\right]Atan\left(\sqrt{\frac{s}{h}}\right)\Big\}\\ &+\frac{6w}{5Gb}\left\{\frac{L^2+a^2-c^2}{2Lh}+\frac{a^2}{2Lu}ln\left(\frac{h+u}{h}\right)-\frac{c^2}{2Ls}ln\left(\frac{h+s}{h}\right)-\frac{a^2}{L\sqrt{hu}}Atan\left(\sqrt{\frac{u}{h}}\right)\right.\\ &+\left.\frac{c^2}{L\sqrt{hs}}Atan\left(\sqrt{\frac{s}{h}}\right)\right\},\end{aligned} \quad (18)$$

$$\begin{aligned}C_1 &= \frac{12}{Eb}\left\{\frac{R_A[4s(L-c)^2(L+2c)+3c^2h(L+c)]}{24Lh^3s}\right.\\ &-\frac{w(L-c)[2s(L-c)^2(L+3c)+3c^2h(L+3c)]}{48Lh^3s}-\frac{M_{AB}(L^2-c^2)}{2Lh^3}\\ &+\frac{wa^3(3h+2u)(4L-3a)}{48Lh^3u}-\frac{R_Aa^2[6L(h+2u)-a(3h+8u)]}{24Lh^3u}\\ &-\frac{M_{AB}a[a(3h+4u)-8L(h+u)]}{8Lh^3(h+u)}+\frac{c^2(wL^2h-2R_ALh-2M_{AB}s)}{16Lh^2s(h+s)}\\ &+\left[\frac{a^3h(3wa-2R_A)-3a^2u(2R_Aa-wa^2-2M_{AB})}{16Lh^2u\sqrt{hu}}\right]Atan\left(\sqrt{\frac{u}{h}}\right)\\ &+\left[\frac{c^3h\{w(2L-3c)-2R_A\}+3c^2s[2R_A(L-c)-w(L-c)^2-2M_{AB}]}{16Lh^2s\sqrt{hs}}\right]Atan\left(\sqrt{\frac{s}{h}}\right)\Big\}\\ &+\frac{6w}{5Gb}\left\{\frac{(L-a)^2-c^2}{2Lh}+\frac{a^2}{2Lu}ln\left(\frac{h+u}{h}\right)-\frac{c^2}{2Ls}ln\left(\frac{h+s}{h}\right)-\frac{a^2}{L\sqrt{hu}}Atan\left(\sqrt{\frac{u}{h}}\right)\right.\\ &+\left.\frac{c^2}{L\sqrt{hs}}Atan\left(\sqrt{\frac{s}{h}}\right)\right\}.\end{aligned} \quad (19)$$

3. Verificación del modelo propuesto

Una manera de verificar el modelo propuesto es como siguiente:

1. Sustituyendo el valor de "M_{AB} = 0, R_A = wL/2, a = c, u = s y dy/dx = 0" para vigas de sección transversal rectangular simplemente apoyadas en la ecuación (10) se obtiene "x = L/2", es decir, cuando la rotación es cero, se produce el desplazamiento máximo (viga simétrica).

2. Sustituyendo el valor de "x = a" en la ecuación (8) y (10), los valores obtenidos de las dos ecuaciones son iguales, es decir, la continuidad está garantizada en este punto para las rotaciones.

3. Sustituyendo el valor de "x = L – c" en la ecuación (10) y (12), los valores obtenidos de las dos ecuaciones son iguales, es decir, la continuidad está garantizada en este punto para las rotaciones.

4. Sustituyendo el valor de "M_{AB} = 0, R_A = wL/2, a = 0, c = 0 y x = L/2" para vigas simplemente apoyadas de sección transversal rectangular constante en la ecuación (11) se obtiene el desplazamiento máximo "$y_{max} = wL^2(24Eh^2 + 25GL^2)/160bh^3EG = 5wL^4/384EI(1 + 48EI/5GA_sL^2)$", presentada por Timoshenko y

Tabla 2. Viga simplemente apoyada

h/L	c/L	s/h	Factores de las rotaciones en el apoyo A y en el apoyo B						Factores del desplazamiento máximo y su ubicación a partir del apoyo A					
			β_{AB}			β_{BA}			ε			ρ		
			MP	MA	MP/MA	MP	MA	MP/MA	MP	MA	MP/MA	MP	MA	MP/MA
0.1								a/L = 0.2; u/h = 1						
	0.2	0.4	483.43	470.23	1.0281	-495.60	-481.86	1.0285	0.5009	0.5009	1.0000	-157.6912	-154.1430	1.0230
		0.6	482.88	469.70	1.0281	-490.16	-476.64	1.0284	0.5006	0.5006	1.0000	-157.4166	-153.8753	1.0230
		1.0	482.00	468.84	1.0281	-482.00	-468.84	1.0281	0.5000	0.5000	1.0000	-156.9742	-153.4447	1.0230
		1.5	481.16	468.02	1.0281	-474.85	-462.02	1.0278	0.4994	0.4994	1.0000	-156.5530	-153.0354	1.0230
		2.0	480.50	467.38	1.0281	-469.64	-457.07	1.0275	0.4990	0.4990	1.0000	-156.2247	-152.7170	1.0230
								a/L = 0.5; u/h = 1						
	0.5	0.4	363.88	352.34	1.0328	-405.13	-392.52	1.0321	0.5117	0.5117	1.0000	-132.6755	-129.3991	1.0253
		0.6	357.01	345.56	1.0331	-381.10	-369.00	1.0328	0.5072	0.5072	1.0000	-129.1762	-125.9432	1.0257
		1.0	346.07	334.76	1.0338	-346.07	-334.76	1.0338	0.5000	0.5000	1.0000	-123.6656	-120.5061	1.0262
		1.5	335.80	324.64	1.0344	-316.45	-305.89	1.0345	0.4933	0.4932	1.0002	-118.5626	-115.4775	1.0267
		2.0	327.89	316.86	1.0348	-295.61	-285.61	1.0350	0.4881	0.4881	1.0000	-114.6835	-111.6594	1.0271
0.2								a/L = 0.2; u/h = 1						
	0.2	0.4	65.38	58.78	1.1123	-67.10	-60.23	1.1141	0.5010	0.5009	1.0002	-21.0419	-19.2679	1.0921
		0.6	65.31	58.71	1.1124	-66.34	-59.58	1.1135	0.5006	0.5006	1.0000	-21.0050	-19.2344	1.0921
		1.0	65.19	58.60	1.1125	-65.19	-58.60	1.1125	0.5000	0.5000	1.0000	-20.9454	-19.1806	1.0920
		1.5	65.07	58.50	1.1123	-64.17	-57.75	1.1112	0.4994	0.4994	0.9998	-20.8882	-19.1294	1.0919
		2.0	64.98	58.42	1.1123	-63.42	-57.13	1.1101	0.4990	0.4990	1.0000	-20.8435	-19.0896	1.0919
								a/L = 0.5; u/h = 1						
	0.5	0.4	49.81	44.04	1.1310	-55.37	-49.07	1.1284	0.5115	0.5117	0.9996	-17.8131	-16.1749	1.1013
		0.6	48.92	43.19	1.1327	-52.18	-46.12	1.1314	0.5070	0.5072	0.9996	-17.3594	-15.7429	1.1027
		1.0	47.50	41.85	1.1350	-47.50	-41.85	1.1350	0.5000	0.5000	1.0000	-16.6430	-15.0633	1.1049
		1.5	46.16	40.58	1.1375	-43.52	-38.24	1.1381	0.4934	0.4932	1.0004	-15.9772	-14.4347	1.1069
		2.0	45.13	39.61	1.1394	-40.70	-35.70	1.1401	0.4882	0.4881	1.0002	-15.4695	-13.9574	1.1083

Gere [24, 25], para sección transversal rectangular constante (las deformaciones por flexión y cortante se consideran).

5. Sustituyendo el valor de “x = 0” en la ecuación (9) se obtiene el desplazamiento cero en el apoyo A.

6. Sustituyendo el valor de “x = L” en la ecuación (13) se obtiene el desplazamiento cero en el apoyo B.

7. Sustituyendo el valor de "x = a" en la ecuación (9) y (11), los valores obtenidos de las dos ecuaciones son iguales, es decir, la continuidad está garantizada en este punto para los desplazamientos.

8. Sustituyendo el valor de "x = L – c" en la ecuación (11) y (13), los valores obtenidos de las dos ecuaciones son iguales, es decir, la continuidad está garantizada en este punto para los desplazamientos.

9. Si se desprecian las deformaciones por cortante, las condiciones anteriores también se verifican y la flecha máxima para la condición de simetría es: “$y_{max} = 5wL^4/32bh3E = 5wL^4/384EI$”, para sección transversal constante (las deformaciones por flexión se consideran únicamente).

10. Si se desprecian las deformaciones por cortante y el valor de “$M_{AB} = wL^2/12$, $R_A = wL/2$ y $a = c = 0$” para vigas rectangulares con apoyos fijos en sus extremos de sección transversal constante se sustituyen en la ecuación (8) se obtiene “$dy/dx = 0$”, es decir, la rotación en el soporte A es cero.

11. Si se desprecian las deformaciones por cortante y el valor de “$M_{AB} = wL^2/12$, $R_A = wL/2$ y $a = c = 0$” para vigas rectangulares con apoyos fijos en sus extremos de sección transversal constante se sustituyen en la ecuación (12) se obtiene “$dy/dx = 0$”, es decir, la rotación en el soporte B es cero.

Por lo tanto, el modelo propuesto en este artículo de la curva elástica para vigas de sección transversal rectangular con cartelas parabólicas bajo cargas uniformemente distribuidas y momentos en sus extremos considerando las deformaciones por flexión y cortante (teoría de Timoshenko) es válido.

Tabla 3. Viga empotrada en sus dos extremos

h/L	c/L	s/h	Factores de momentos de empotramiento y reacciones en el apoyo A						Factores del desplazamiento máximo y su ubicación a partir del apoyo A					
			m_{AB}			α_{AB}			ε			ρ		
			MP	MA	MP/MA	MP	MA	MP/MA	MP	MA	MP/MA	MP	MA	MP/MA
0.1			a/L = 0.2; u/h = 1											
	0.2	0.4	0.1020	0.1022	0.9980	0.5177	0.5181	0.9992	0.5159	0.5174	0.9971	-23.0340	-19.4949	1.1815
		0.6	0.0994	0.0995	0.9990	0.5106	0.5108	0.9996	0.5096	0.5106	0.9980	-21.8765	-18.3382	1.1929
		1.0	0.0955	0.0955	1.0000	0.5000	0.5000	1.0000	0.5000	0.5000	1.0000	-20.2782	-16.7487	1.2107
		1.5	0.0923	0.0922	1.0011	0.4910	0.4908	1.0004	0.4915	0.4906	1.0018	-18.8692	-15.3540	1.2289
		2.0	0.0900	0.0899	1.0011	0.4846	0.4843	1.0006	0.4853	0.4837	1.0033	-17.9613	-14.3894	1.2482
			a/L = 0.5; u/h = 1											
	0.5	0.4	0.1167	0.1171	0.9966	0.5352	0.5360	0.9985	0.5268	0.5293	0.9953	-17.5394	-14.2959	1.2269
		0.6	0.1113	0.1116	0.9973	0.5221	0.5227	0.9989	0.5166	0.5184	0.9965	-16.0968	-12.9248	1.2454
		1.0	0.1025	0.1025	1.0000	0.5000	0.5000	1.0000	0.5000	0.5000	1.0000	-13.8806	-10.7211	1.2947
		1.5	0.0941	0.0937	1.0043	0.4782	0.4774	1.0017	0.4840	0.4819	1.0044	-11.9922	-8.9222	1.3441
		2.0	0.0877	0.0870	1.0080	0.4610	0.4595	1.0033	0.4713	0.4672	1.0088	-10.6353	-7.6098	1.3976
0.2			a/L = 0.2; u/h = 1											
	0.2	0.4	0.1016	0.1022	0.9941	0.5169	0.5181	0.9977	0.5126	0.5174	0.9907	-4.2076	-2.4369	1.7266
		0.6	0.0992	0.0995	0.9970	0.5102	0.5108	0.9988	0.5077	0.5106	0.9943	-4.0618	-2.2923	1.7719
		1.0	0.0955	0.0955	1.0000	0.5000	0.5000	1.0000	0.5000	0.5000	1.0000	-3.8583	-2.0936	1.8429
		1.5	0.0925	0.0922	1.0033	0.4913	0.4908	1.0010	0.4933	0.4906	1.0055	-3.6684	-1.9193	1.9113
		2.0	0.0903	0.0899	1.0044	0.4852	0.4843	1.0019	0.4885	0.4837	1.0099	-3.5590	-1.7987	1.9787
			a/L = 0.5; u/h = 1											
	0.5	0.4	0.1156	0.1171	0.9872	0.5333	0.5360	0.9950	0.5217	0.5293	0.9856	-3.4339	-1.7870	1.9216
		0.6	0.1106	0.1116	0.9910	0.5208	0.5227	0.9964	0.5133	0.5183	0.9904	-3.2280	-1.6097	2.0053
		1.0	0.1025	0.1025	1.0000	0.5000	0.5000	1.0000	0.5000	0.5000	1.0000	-2.9199	-1.3401	2.1789
		1.5	0.0948	0.0937	1.0117	0.4797	0.4774	1.0048	0.4876	0.4819	1.0118	-2.6590	-1.1153	2.3841
		2.0	0.0891	0.0870	1.0241	0.4640	0.4595	1.0098	0.4783	0.4672	1.0238	-2.4646	-0.9512	2.5910

4. Resultados

La Tabla 2 muestra la comparación de los dos modelos para obtener los factores para las rotaciones en los apoyos y los desplazamientos máximos para una viga simplemente apoyada, el modelo propuesto (MP) es el modelo matemático presentado en este trabajo tomando en cuenta las deformaciones por cortante y flexión, y el modelo actual (MA) considera únicamente las deformaciones por flexión.

La Tabla 2 presenta para h = 0.1L y h = 0.2L. Estas comparaciones se realizan para G = 5E/12 para concreto, a = 0.2L, 0.5L; u = h; c = 0.2L, 0.5L; s = 0.4h, 0.6h, h, 1.5h, 2h.

dónde: Θ_{AB} (rotación en el apoyo A) = β_{AB}wL/Eb; Θ_{BA} (rotación en el apoyo B) = β_{BA}wL/Eb; x_{AB} (ubicación del momento máximo a partir del apoyo A) = εL; y_{max} (desplazamiento máximo) = ρwL/Eb.

Tal como se muestra en la Tabla 2, los factores en las rotaciones para los apoyos “A” y “B” están influenciados por la altura “h”.

A medida que aumenta la altura de las cartelas en el apoyo “B” se observa una disminución (valor absoluto) de estos factores para el mismo apoyo “B” y en el apoyo “A” se produce una disminución, esto es para los dos modelos.

Además, los factores en los desplazamientos máximos están influenciados por la altura “h”. A medida que aumenta la altura de las cartelas en el apoyo “B” se observa una disminución de estos factores, esto es para los dos modelos.

De acuerdo a los resultados, el modelo propuesto es mayor en todos los casos para las rotaciones en los apoyos y los desplazamientos máximos, y para los dos casos en h = 0.2L, donde la mayor diferencia es de 14.01% para la rotación en el apoyo “B”, y para los desplazamientos máximos es de 10.83%.

ISSN 2007-9737

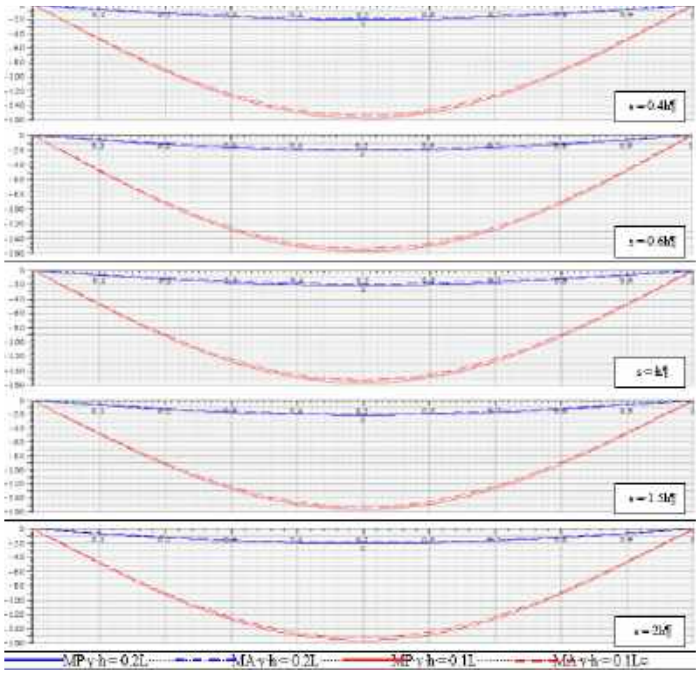


Fig. 3. Deflexiones en vigas simplemente apoyadas para a = 0.2L, c = 0.2L y u = h

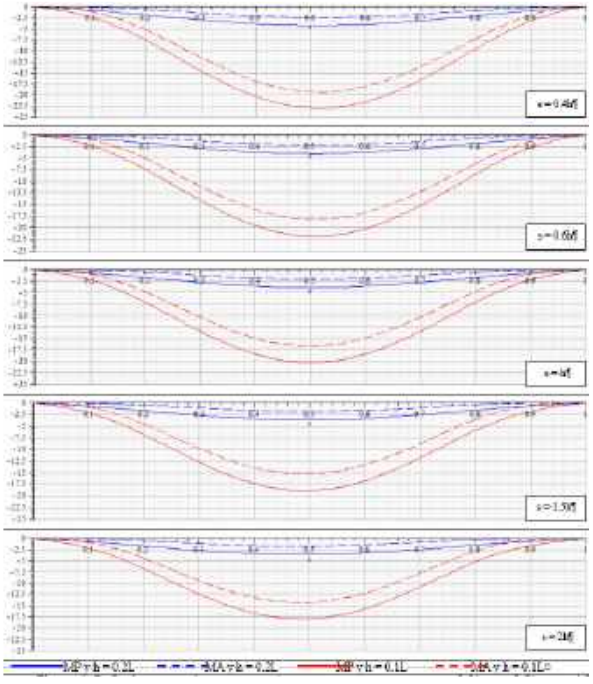


Fig. 4. Deflexiones en vigas empotradas en sus extremos para a = 0.2L, c = 0.2L y u = h

La Tabla 3 presenta la comparación de los dos modelos para encontrar los factores de los momentos y reacciones en el apoyo A y los desplazamientos máximos para una viga

empotrada en ambos extremos. La Tabla 3 presenta h = 0.1L y h = 0.2L. Estas comparaciones se realizan para G = 5E/12 para concreto, a = 0.2L, a = 0.5L; u/h = 1.0; c = 0.2L, 0.5L; s = 0.4h, 0.6h, h, 1.5h, 2h.

La Tabla 3 se obtienen sustituyendo G = 5E/12; a = 0.2L, 0.5L; u = h; c = 0.2L, c = 0.5L; s = 0.4h, 0.6h, h, 1.5h, 2h; h = 0.1L, 0.2L; x = 0 en la ecuación (8) y x = L en la ecuación (12) y estas ecuaciones se hacen iguales a cero. Posteriormente, estas dos ecuaciones se resuelven para obtener el momento de empotramiento en el apoyo "A" "$M_{AB} = m_{AB}wL^2$" y la reacción en el apoyo "A" "$R_A = \alpha_{AB}wL$". El momento de empotramiento en el apoyo "B" "$M_{BA} = m_{BA}wL^2$" y la reacción en el apoyo "B" "$R_B = \alpha_{BA}wL$" se obtienen por equilibrio estático.

dónde: M_{AB} (momento de empotramiento en el apoyo A) = $m_{AB}wL^2$; R_{AB} (reacción en el apoyo A) = $\alpha_{AB}wL$.

Tal como se presenta en la Tabla 3, los factores para los momentos y reacciones en el apoyo "A" están influenciados por la altura "h". A medida que se aumenta la altura de las cartelas en el apoyo "B" se observa una disminución de estos factores para el apoyo "A" en ambos factores, esto es para los dos modelos. Además, los factores en los desplazamientos máximos están influenciados por la altura "h".

A medida que se aumenta la altura de las cartelas en el apoyo "B" se observa una disminución (valor absoluto) de estos factores, esto es para los dos modelos. De acuerdo a los resultados, el modelo propuesto para los momentos y para las reacciones en el apoyo "A" es menor en s = 0.4h, 0.6h; igual en s = h; y mayor en s = 1.5h, 2h, y los desplazamientos máximos son mayores, y para los dos casos en h = 0.2L, donde la mayor diferencia es de 2.41 % para los momentos, para las reacciones en el apoyo "A" es de 1.0098 veces el modelo propuesto con respecto al modelo actual, y para los desplazamientos máximos es de 2.5910 veces el modelo propuesto con respecto al modelo actual.

La Figura 3 muestra la curva elástica para una viga rectangular simplemente apoyada con cartelas parabólicas bajo carga uniformemente distribuida para valores de G = 5E/12; a = 0.2L; u = h; c = 0.2L; s = 0.4h, 0.6h, h, 1.5h, 2h; h = 0.1L, 0.2L por el modelo propuesto "MP" y el modelo actual "MA".

La Figura 4 muestra la curva elástica para una viga rectangular empotrada en sus extremos con cartelas parabólicas bajo carga uniformemente distribuida para valores de G = 5E/12; a = 0.2L; u = h; c = 0.2L; s = 0.4h, 0.6h, h, 1.5h, 2h; h = 0.1L, 0.2L por el modelo propuesto "MP" y el modelo actual "MA".

Figura 4. Deflexiones en vigas empotradas en sus extremos para a = 0.2L, c = 0.2L y u = h

Los valores de la curva elástica (deflexiones) para la viga simplemente apoyada es mayor para el modelo propuesto con respecto al modelo actual para h = 0.1L y h = 0.2L, y los valores mayores se presentan para h = 0.1L (ver Figura 3). Para la viga empotrada en ambos extremos es mayor para el modelo propuesto con respecto al modelo actual para h = 0.1L y h = 0.2L, y los valores mayores son para h = 0.1L (ver Figura 4).

5 Conclusiones

El modelo propuesto para obtener las rotaciones y los desplazamientos en cualquier lugar de la viga de sección transversal rectangular con cartelas parabólicas bajo una carga uniformemente distribuida y momentos en los extremos considerando las deformaciones por flexión y cortante (teoría de Timoshenko) ha sido desarrollado para el caso general.

La técnica matemática presentada en esta investigación es adecuada para encontrar las rotaciones y las deflexiones en cualquier lugar de la viga sometida a una carga uniformemente distribuida y cualquier tipo de momentos aplicados en sus extremos, debido a que se muestran las fórmulas matemáticas de la curva elástica.

Las principales conclusiones son:

1 El modelo actual no está influenciado por la relación "h/L" para las vigas empotradas en sus extremos para los momentos y reacciones en el apoyo "A", y la ubicación de los desplazamientos máximos, y también para las vigas simplemente apoyadas en la ubicación de los desplazamientos máximos.
2 La mayor diferencia se presenta en "h = 0.20L" con respecto a "h = 0.10L", es decir, a

mayor relación de “h/L” aparece mayor diferencia para las vigas simplemente apoyadas en las rotaciones para los apoyos “A” y “B”, y los desplazamientos máximos, y para las vigas empotradas en ambos extremos para los momentos y reacciones en el apoyo “A”, y los desplazamientos máximos.

3 El modelo propuesto para las vigas simplemente apoyadas es mayor en todos los casos para las rotaciones en los apoyos y para los desplazamientos máximos con respecto al modelo actual.

4 El modelo propuesto para las vigas empotradas en ambos extremos es menor en s = 0.4h, 0.6h; igual en s = h; y mayor en s = 1.5h, 2h para los momentos y para las reacciones en el apoyo “A”, y para los desplazamientos máximos son mayores con respecto al modelo actual.

5 Los factores de los momentos de empotramiento se pueden verificar en el documento propuesto por Velázquez-Santillán [13].

Los desplazamientos máximos por el modelo propuesto (las deformaciones por flexión y cortante se consideran) son mayores para las vigas simplemente apoyadas, y para las vigas empotradas en ambos extremos respecto al modelo actual (las deformaciones por flexión se consideran).

Por lo tanto, las deflexiones máximas que actúan sobre las vigas del modelo propuesto en este trabajo deben compararse con las deflexiones máximas permitidas por los códigos de construcción, porque en algunas condiciones podría ser que no cumpla con los estándares establecidos por los códigos de construcción.

Entonces, el modelo propuesto es más apropiado y seguro con respecto al modelo actual para el análisis estructural, debido a que los esfuerzos cortantes y los momentos flectores están presentes en cualquier tipo de estructura y aparecen deformaciones por flexión y cortante.

Las sugerencias para futuras investigaciones pueden ser: 1) Cuando la carga aplicada es diferente a una carga uniformemente distribuida; 2) Cuando la sección transversal de la viga sea diferente a una rectangular.

Agradecimientos

La investigación descrita en este trabajo fue financiada por el Instituto de Investigaciones Multidisciplinarias de la Facultad de Contaduría y Administración de la Universidad Autónoma de Coahuila. Los autores también agradecen a los revisores y al editor por los comentarios y sugerencias para mejorar la presentación. El estudiante de doctorado Gilberto Ramírez Muñoz (CVU: 547985) agradece al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico.

Referencias

1. **Lee, K. (2002).** Large deflections of cantilever beams of non-linear elastic material under a combined loading. International Journal of Non-Linear Mechanics, Vol. 37, No. 3, pp. 439–443. DOI: 10.1016/s0020-7462(01) 00019-1.
2. **Dado, M., Al-Sadder, S. (2005).** A new technique for large deflection analysis of non-prismatic cantilever beams. Mechanics Research Communications, Vol. 32, No. 6, pp. 692–703. DOI: 10.1016/j.mechrescom. 2005.01.004.
3. **Borboni, A., De-Santis, D. (2014).** Large deflection of a non-linear, elastic, asymmetric Ludwick cantilever beam subjected to horizontal force, vertical force and bending torque at the free end. Meccanica, Vol. 49, No. 6, pp. 1327–1336. DOI: 10.1007/s11012-014-9895-z.
4. **Banerjee, A., Bhattacharya, B., Mallik, A. K. (2008).** Large deflection of cantilever beams with geometric non-linearity: Analytical and numerical approaches. International Journal of Non-Linear Mechanics, Vol. 43, No. 5, pp. 366–376. DOI: 10.1016/j.ijnonlinmec.2007.12.020.
5. **Solano-Carrillo, E. (2009).** Semi-exact solutions for large deflections of cantilever beams of non-linear elastic behavior. International Journal of Non-Linear Mechanics, Vol. 44, No. 2, pp. 253–256. DOI: 10.1016/j.ijnonlinmec.2008.11.007.

6. **Chen, L. (2010).** An integral approach for large deflection cantilever beams. International Journal of Non-Linear Mechanics, Vol. 45, No. 3, pp. 301–305. DOI: 10.1016/j.ijnonlinmec.2009.12.004.
7. **Yau, J. D. (2020).** Closed-form solution of large deflection for a guyed cantilever column pulled by an inclination cable. Journal of Marine Science and Technology, Vol. 18, No. 1, pp. 130–136. DOI: 10.51400/2709-699 8.1874.
8. **Brojan, M., Cebron, M., Kosel, F. (2012).** Large deflections of non-prismatic non-linearly elastic cantilever beams subjected to non-uniform continuous load and a concentrated load at the free end. Acta Mechanica Sinica, Vol. 28, No. 3, pp. 863–869. DOI: 10.1007/s10409-012-0053-3.
9. **Luévanos-Rojas, A. (2014).** A mathematical model of elastic curve for simply supported beams subjected to a uniformly distributed load taking into account the shear deformations. ICIC Express Letters Part B: Applications, Vol. 5, No. 3, pp. 885–890.
10. **Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M., Kalashnikov, V. V. (2016).** A mathematical model of elastic curve for simply supported beams subjected to a concentrated load taking into account the shear deformations. International Journal of Innovative Computing, Information and Control, Vol. 12, No. 1, pp. 41–54. DOI: 10.24507/ijicic.12.01.41.
11. **Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2016).** Modeling for mechanical elements of rectangular members with straight haunches using software: Part 1. International Journal of Innovative Computing, Information and Control, Vol. 12, No. 3, pp. 973–985. DOI: 10.24507/ijicic.12.03.959.
12. **Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2016).** Modeling for mechanical elements of rectangular members with straight haunches using software: Part 2. International Journal of Innovative Computing, Information and Control, Vol. 12, No. 4, pp. 1027–1041. DOI: 10.24507/ijicic.12.04.1027.
13. **Velázquez-Santillán, F., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2019).** Modelado para trabes de sección transversal rectangular con cartelas parabólicas: Parte 1. Computación y Sistemas, Vol. 23, No. 2, pp. 557–568. DOI: 10.13053/CyS-23-2-2872.
14. **Sandoval-Rivas, R., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2019).** Modelado para trabes de sección transversal rectangular con cartelas parabólicas: Parte 2. Computación y Sistemas, Vol. 23, No. 3, pp. 1115–1124. DOI: 10.13053/CyS-23-3-2873.
15. **Montano-Perez, B. E., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M., Jaramillo-Rosales, M. (2020).** Design aids for rectangular cross-section beams with straight haunches: Part 1. International Journal of Innovative Computing, Information and Control, Vol. 16, No. 6, pp. 1915–1928. DOI: 10.24507/ijicic.16.06.1915.
16. **Gaona-Tamez, L. L., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M., Jaramillo-Rosales, M. (2020).** Design aids for rectangular cross-section beams with straight haunches: Part 2. International Journal of Innovative Computing, Information and Control, Vol. 16, No. 6, pp. 1929–1942. DOI: 10.24507/ijicic.16.06.1929.
17. **Crispín-Herrera, C. Y., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2021).** Ayudas de diseño para trabes de sección transversal rectangular con cartelas parabólicas: Parte 1. Computación y Sistemas, Vol. 25, No. 3. DOI: 10.13053/cys-25-3-3777.
18. **Luévanos-Soto, R. M., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2021).** Ayudas de diseño para trabes de sección transversal rectangular con cartelas parabólicas: Parte 2. Computación y Sistemas, Vol. 25, No. 4. DOI: 10.13053/CyS-25-4-3778.
19. **Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M., Kalashnikov, V. V. (2020).** Optimal design of reinforced concrete beams for rectangular sections with straight

haunches. Revista de la Construcción, Vol. 19, No. 1. DOI: 10.7764/RDLC.19.1.90-102.

20. **García-Canales, E., Luévanos-Rojas, A., López-Chavarría, S., Medina-Elizondo, M. (2020).** Costo mínimo para trabes rectangulares de concreto reforzado con cartelas parabólicas. Computación y Sistemas, Vol. 24, No. 3. DOI: 10.13053/CyS-24-3-3306.
21. **Majumder, G., Kumar, K. (2013).** Deflection and stress analysis of a simply supported beam and its validation using ANSYS. International Journal of Mechanical Engineering and Computer Applications, Vol. 1, No. 5, pp. 17–20.
22. **Debnath, V., Debnath, B. (2014).** Deflection and stress analysis of a beam on different elements using ANSYS APDL. International Journal of Mechanical Engineering and Technology, Vol. 5, No. 6, pp. 70–79.
23. **Sihua, D., Ze, Q., Li, W. (2015).** Nonlinear analysis of reinforced concrete beam bending failure experimentation based on ABAQUS. International Conference on Information Sciences, Machinery, Materials and Energy, Published by Atlantis Press, pp. 440–444. DOI: 10.2991/icismme-15.2015.88.
24. **Timoshenko, S. P. (1947).** Strength of materials: Part I. Elementary Theory and Problems, 2nd Edition, Van Nostrand Company, New York.
25. **Timoshenko, S. P., Gere, J. M. (1972).** Mechanics of materials, Van Nostrand Reinhold, New York.

Article received on 23/03/2022; accepted on 08/04/2024.
˚Corresponding author is Arnulfo Luévanos Rojas.

ISSN 2007-9737

Comparing Pre-Trained Language Model for Arabic Hate Speech Detection

Kheir Eddine Daouadi[1,*], Yaakoub Boualleg[1], Oussama Guehairia[2]

[1] Echahid Cheikh Larbi Tebessi University,
Laboratory of Vision and Artificial Intelligence,
Algeria

[2] Mohamed Khider University of Biskra,
Faculty of Sciences and Technology,
Algeria

{kheireddine.daouadi, yaakoub.boualleg}@univ-tebessa.dz, oussama.guehairia@univ-biskra.dz

Abstract. Today, the classification of hate speech in Arabic tweets has garnered significant attention from scholars worldwide. Although numerous classification approaches proposed in response to this interest, two primary challenges persist are reliance on handcrafted features and limited performance rates. This paper addresses the task of identifying Arabic hate speech on Twitter, aiming to deepen insights into the efficacy of novel machine-learning techniques. Specifically, we compare the performance of traditional machine learning-based approaches with state-of-the-art pre-trained language models based on Transfer Learning, as well as deep learning models. Our experiments, conducted on a benchmark dataset using a standard evaluation scenario, reveal several key findings. Firstly, multidialectal pre-trained language models demonstrate superior performance compared to monolingual and multilingual variants. Secondly, fine-tuning the pre-trained large language models significantly enhances the accuracy of hate speech classification in Arabic tweets. Our primary contribution lies in achieving promising results for the corresponding task through the application of multidialectal pre-trained language models trained on Twitter data.

Keywords. Arabic hate speech detection, fine-tuning, transfer learning, AraBERT.

1. Introduction

Nowadays, hate speech has garnered significant attention from scholars worldwide. Originally, this form of content was shared via conventional media outlets.

However, the global availability of the Internet, facilitated by social media like Twitter, YouTube, and Facebook, has led to an exponential increase in users expressing their opinions and sharing posts. Regrettably, these posts can occasionally exert adverse psychological impacts on social media users, with extreme cases even resulting in instances of suicide [2]. The proliferation of unregulated text on social media represents a concerning phenomenon, particularly when such content contains hate speech. The European Union has adopted a legislative approach to address this issue.

Specifically, the Commission of the European Union has exerted pressure on numerous social media platforms to adopt a hate speech code. As part of this code, platforms have committed to reviewing the 'notifications for elimination of hate speech' within a 24-hour and facilitating direct notification to law enforcement agencies.

However, fulfilling this pledge proves challenging owing to the missed of clarity regarding the precise scope of hate speech, stemming from inadequate data collection and systematic reporting mechanisms.

Consequently, platforms often rely on their user communities to identify and report instances of hateful speech.

This task poses significant complexity for social media platforms. Given the vast volume of data shared daily, coupled with the absence of efficient automated systems, the community of natural

language processing is motivated to undertake research into hate speech detection.

Additionally, there are a significant demand for study focused on language than English [3]. Today, researchers are leveraging Twitter data to propose various approaches for Arabic hate speech classification.

However, two primary challenges persist reliance on handcrafted features and limited performance rates. Automatic classification of Arabic hate speech using conventional learning algorithms like Support Vector Machine (SVM), and Naïve Bayes (NB) has demonstrated acceptable results.

Nevertheless, they rely on handcrafted features derived using pre-defined methods like Term Frequency-Inverse Document Frequency (TF-IDF), Bag of Word (BoW), and Term Frequency (TF). Recently, Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and Convolution Neural Network (CNN) have already shown promising results. However, they depend on some pre-defined word embedding models like AraBERT, Mazajk, and AraVec.

This paper offers a comparative examination of various machine-learning methodologies for the classification of Arabic hate speech on Twitter. We evaluate the classification models using a benchmark dataset that contains tweets annotated for hate speech classification.

The major contributions of this work are briefly noted as follows:

- We evaluate three suggested DL-based approaches (Bi-LSTM, LSTM, and CNN) along with traditional machine learning models (SVM and NB).
- We compare the accuracy results of the recent pre-trained language model utilizing transformer mechanisms. Including multi-lingual ones (XLM and BERT), a mono-lingual model (AraBERT), and a multi-dialectal model (AraBERT-Twitter).
- We compare the performance of the transformers-based model with our baseline.

The rest of this manuscript is structured as follows. Section 2 presents the related works. In Section 3, we discuss the data and methodology. In Section 4, we focus on the experiments and evaluation results. In Section 5, we discuss our main contribution. In Section 6, we conclude the paper.

Table 1. Overview of our interested dataset

Parameters	Value
Tweet counts	11634
Words Counts	138.3 K
Unique words	37.9 K
Average words per tweet	11.9

2. Related Works

The emergence of the Twitter platform has encouraged a multitude of research avenues including topic detection [4], organization detection [5, 6], and bot detection [6, 7]. Thanks to their importance, Arabic hate speech classification has garnered significant attention from scholars worldwide.

Numerous methods and systems have been suggested to tackle this challenging classification task. They follow two major approaches: a traditional based approach and a deep learning-based approach.

2.1 Traditional Approaches

In this scenario, conventional classification methods depend on feature engineering, wherein texts are transformed into feature vectors before classification using standard algorithms like SVM and NB. Examples of conventional approaches are outlined briefly. The authors in [8] underscore the significance of utilizing datasets from multiple platforms to enhance the generalizability of the classifier in detecting offensive language.

They experimented with SVM and TF-IDF, and achieved F1 score of 84%. Besides, authors in [9] explore the influence of preprocessing steps on offensive language and hate speech classification. They demonstrate that thorough preprocessing techniques have notable effects on detection rates. The best experimental outcomes were achieved using SVM and BoW, attaining F1 scores of 95%

Table 2. Optimized values of hyperparameters explored in DL models

Parameters	CNN	LSTM	BiLSTM
Size	100	75	75
Dropout	0.25	0.25	0.5
Activation	tanh	relu	relu
Optimizer	Adam	Adam	Adam
Batch size	8	32	64
Learning rate	0.01	0.002	0.002

and 89% for hate speech and offensive language, respectively.

Likewise, authors in [10] use the Arabert embedding with Deep Forest, the best experimental results showed acceptable macro-averaged and weighted-average F1-score results of 63% and 80%, respectively.

Furthermore, authors in [11] employ BoW and TF-IDF to categorize tweets as offensive or normal. Their findings indicate that ensemble classifier (Bagging) outperforms single classifier, achieving F1 score of 0.88. In similar work, the authors in [12] categorize tweets into those of normal, hate, and abusive.

They utilize NB and SVM classifiers with trigrams, bigrams, and unigrams. The best experimental outcomes are achieved with NB, resulting F1 score of 0.896 and 0.744% for (abusive and hate vs. normal) and (normal vs. hate vs. abusive) tasks, respectively.

In a distinct approach, researchers in [13] use Social Graph, tweet-based and profile features to differentiate non-abusive Twitter accounts from abusive ones. The best-achieved F1 score was 85% using the NB classifier.

2.2 Deep Learning Approaches

In this context, these methodologies utilize a neural network capable of automatically learning representations of input tweet texts by varying the level of the abstraction. These learned representations are then leveraged to execute the classification task. Commonly employed embedding models include Mazajk and AraVec.

The prevailing DL architectures utilized for Arabic hate speech classification encompass BERT, LSTM, and GRU. Below, we briefly outline some examples of DL approaches. Authors in [14] categorize tweets into religious, general-hate, racial, sexism, or normal.

They employ an embedding layer randomly initialized to learn the word embedding from the training data. The best experimental outcomes are achieved using the Hybrid CNN-LSTM model, resulting in an F1 score of 73%. In a similar study, the authors in [15] evaluate two AraVec models to categorize tweets into normal and hateful. The experiment with Hybrid CNN-LSTM achieved F1 score results of 71.68%.

Likewise, authors in [16] explore the influence of word embedding and neural networks on the performance rates across various classification tasks. They train multiple embedding models and subsequently employ these models to train several neural networks for different classification task.

The best experimental outcomes are observed with Skip-gram and CNN, resulting in F1 scores of 70.80%, 75.16%, and 87.22% for the 6-class, 3-class, and 2-class classifications, respectively. Besides, the authors in [18] use ensemble CNN and Bidirectional LSTM (BiLSTM) classifiers based on the AraBERT. The best outcome is obtained using the average-based ensemble approach, yielded F1 score of 80.23%(BiLSTMs), 84.01% (CNNs), and 91.12% (CNNs) for 6 class, 3 class, and binary classification tasks, respectively.

In a similar, authors in [17] use AraVec and AraBERT to categorize tweets as being normal, abusive, or hateful. The best performance was achieved using CNN, yielding F1 score of 0.721. In a similar, the authors in [19] use CNN with Multilingual BERT embedding model, yielding F1 score of 75.51%, 78.9%, and 87.03% for 6 class, 3 class, and binary classification tasks, respectively.

Furthermore, authors in [20] utilize a bidirectional GRU enhanced by an attention layer alongside the AraVec to identify offensive language and hate speech. Moreover, they examine the effect of different oversampling techniques and pre-processing techniques on the performance results.

The best outcomes consist of F1 score results of 0.859% and 0.75 for offensive and hate speech, respectively.

In a distinct strategy, researchers in [21] fine-tune the pre-trained AraBERT [47] for classifying

tweets being offensive, vulgar, hate speech, or clean. They achieved an F1 score result of 83.2%.

2.3 Gaps and Contributions

After reviewing the current studies, we can realize that some current pre-trained language models have not yet been evaluated for hate speech classification from Arabic tweets. Moreover, there is no existing study where Monolingual, Multilanguage, and Multidialectal pre-trained language models based on transformers are compared to demonstrate their validity for classifying Arabic hate speech. In this work, we rely on transfer learning models due to their major advantages:

They can capture long-term dependency in language, while it does not need a large dataset. Additionally, we conduct a comparative analysis between deep learning approaches and conventional approaches as our baselines.

3. Data and Methodology

2.4 Dataset Description

The dataset used in this paper was published in [14]. The basic characteristics of the corresponding dataset are presented in Table. 1. The tweets were collected using a curated list of hashtags known to elicit hateful content on Twitter. Subsequently, the retrieved tweets were manually annotated. The racial hate speech class constitutes a minor subset of the tweets, whereas the majority belong to the non-hate class.

2.5 Features Representation

The efficacy of a classification system depends on how it represents the text. Specifically, for tasks such as tweet classification, it is essential to convert the tweet's textual content into an appropriate representation for learning a classifier.

Hence, in this work, we adopt three distinct representations, which are outlined briefly below. The Bag of Words (BoW) [22] method stands as one of the foremost techniques employed for information retrieval. BoW centers on counting the occurrences of words within a given text corpus.

Table 3. Optimized hyperparameter for the pre-trained language models (E=Epochs, BS=Batch Size, LR=Learning Rate)

Model	E	BS	LR
xlm-roberta-base	10	16	$3e^{-5}$
xlm-roberta-large	5	64	$1e^{-5}$
bert-base-arabic	3	8	$4e^{-5}$
bert-large-arabic	2	16	$1e^{-5}$
bert-base-arabert	4	8	$2e^{-5}$
bert-large-arabert	5	8	$1e^{-5}$
base-multilingual-cased	5	8	$1e^{-5}$
multi-dialect-bert-base-arabic	4	8	$3e^{-5}$
albert-base-arabic	3	8	$2e^{-5}$
albert-large-arabic	3	8	$1e^{-5}$
base-arabertv02-twitter	4	16	$1e^{-5}$
large-arabertv02-twitter	2	16	$1e^{-5}$

This approach generates a vocabulary comprising unique words found across all tweets and utilizes these as feature vectors to indicate the absence or presence of such words within the vocabulary. Term Frequency Inverse Document Frequency (TF–IDF) weighting scheme that combines Inverse Document Frequency (IDF) with Term Frequency (TF).

This technique is commonly used for Text Mining and Information Retrieval, which converts the tweet to a matrix of integer producing sparse matrices of the counts [23]. Word Embedding (WE) [24, 25] stands as an effective technique that has seen considerable success in recent years. A feature vectors space consists of unsupervised word embedding vectors.

These vectors represent the semantic spaces of each word in a real-valued space. Word embedding vectors offer a dense representation of word meaning, where the word is characterized as a real-valued features vector. Word embedding models can be produced using static pre-trained models like word2vec [25], GloVe [26], and fastText [27], or by employing contextual pre-trained embedding models like BERT [28].

Table 4. F1 score results of both traditional and DL classifiers

Model	Macro	Weighted
NB-BoW	43.47	70.98
SVM-BoW	48.77	73.07
NB-TF_IDF	28.68	65.02
SVM-TF_IDF	48.29	72.88
NB-AraVec	22.97	61.52
SVM- AraVec	38.78	63.29
CNN-AraVec	50.49	73.57
LSTM-AraVec	49.72	73.26
BiLSTM-AraVec	51.54	74.13

2.6 Model Description

This subsection presents the classification models. We outline the conventional learning systems, the deep learning architectures, and the transfer learning models we have used.

Traditional machine learning. We evaluate two models most commonly used the Multinomial Naïve Bayes and Support Vector Machine, which predict classes based on a combination of features.

Support Vector Machine (SVM) [29] most well-known classifiers since it is highly accurate and effective in text classification. This classifier offers the advantage of typically performing well even when trained with a limited amount of data [30]. For the hyperparameter optimization, we experiment with various values: 'C' = [1, 0.01, 10, 0.1], 'class_weight' = [balanced, None], 'penalty' = [l2, l1]. Following the optimization, we utilize the linear SVM classifier with its default configuration.

Multinomial Naïve Bayes (MNB) [31] is one of the most well-known classifiers since it is highly accurate and effective in text classification.

It operates by considering the frequency of such word to generate in a multinomial fashion the data distribution. For the hyper-parameter optimization, we evaluate the values of: 'Alpha' = [0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1] and 'fit_prior' = [True, False]. After optimization, the default parameters of the classifier were used. To derive feature vectors for inclusion for such classifiers, we utilize the three types of tweet representation previously discussed (i.e. Word Embedding, TF-IDF, and BoW).

Deep Learning. The deep learning methods explored in this study are briefly described in the following. Convolution Neural Network (CNN) stands as the most effective neural network model, offering an alternative approach to traditional feedforward neural networks. In CNN architecture, different layers are sparsely connected, linking a local region of an input layer with neurons in the subsequent layer.

The work by [32] was the first that applied CNN to text classification, wherein words are transformed into numerical values via word embedding. A 2-dimensional matrix is formed from the tweet text, where each row is a word vector in that tweet. The typical CNN architecture encompasses several stages, including a fully connected layer, a pooling layer, and a convolutional layer.

Recurrent Neural Network (RNN) [33] is another class of neural network to address the challenge of sequential learning faced by the conventional neural network. The connections among nodes construct the directed graph over the temporal sequences, enabling the model to highlight the dynamic temporal behaviors.

Long Short-Term Memory (LSTM) represents the most widely recognized variant, as introduced by [34], and trained via backpropagation through time. LSTM networks are equipped with memory blocks, enabling them to learn the temporal sequence and their long-terms dependency effectively. On the other hand, Bi-directional Long Short-Terms Memory (BiLSTMs) facilitate the two-way information flow. This architecture involves training two LSTM network simultaneously, one for the forward and one for the reverse direction [35].

In this scenario, we use word embedding as a feature representation. We particularly use Aravec [36], which consists of 300-dimensional vectors for each word.

Transfer Learning. In this scenario, the process is to adjust a pre-trained language model to a new dataset through the transfer of the learned features. In other words, a technique to improve learning of a new task by transferring knowledge from the learned task [37].

The transformer operates as an attention mechanism, enabling the learning of contextual relationships between words within a text. It includes two primary components: the encoder, which processes the textual input; and the decoder generates estimates for the corresponding task [38].

Unlike directional models that sequentially process textual input (e.g., right-to-left or left-to-right), the encoder of the transformer simultaneously processes the entire sequence.

This approach enables model to capture the word context based on their surrounding context as a whole. The authors in [38] achieved an enhancement in the translation task with the use of the attention mechanism avoiding relying on RNN, paving the way for additional transformer architectures.

Bidirectional Encoder Representations from Transformers (BERT) is the first transformer-based language model introduced by Google. The model is pre-trained on large unsupervised text data based on two self-supervision tasks:

Masked Language Modeling and Next Sentence Prediction. In the first task, approximately 0.15 of the words in such sentences were masked at random, and the model forecast the masked words.

The second task involves the classification of two sentences, wherein the model was tasked with discerning the original orders between the two sentences, thereby enhancing document-levels understanding.

Alternatively, the authors in [39] proposed a cross-lingual language model refer as XLM, improving BERT while attaining remarkable achievements across different machine translation and cross-lingual classification tasks.

Unlike BERT, which is not adjusted for multi-lingual tasks due to limited shared vocabulary across languages, XLM tackles this challenge by processing all languages using a shared vocabulary generated based on a preprocessing method called Byte Pair Encoding [40, 41]. Additionally, XLM uses the dual-language training mechanism alongside BERT so as to learn inter-language word relationship effectively.

Table 5. F1 score results of pre-trained language models before fine-tuning

Model	Macro	Weighted
xlm-roberta-base	31.64	67.13
xlm-roberta-large	29.03	65.99
bert-base-arabic	51.90	74.96
bert-large-arabic	52.54	74.82
bert-base-arabert	54.06	75.74
bert-large-arabert	53.98	75.22
base-multilingual-cased	47.12	72.57
multi-dialect-bert-base-arabic	**63.38**	**80.85**
albert-base-arabic	57.75	78.10
albert-large-arabic	59.27	78.59
base-arabertv02-twitter	57.23	77.84
large-arabertv02-twitter	64.07	80.82

3 Experiment and Evaluation

This section outlines the experimental procedures and the evaluation conducted to assess the efficacy of the pre-trained language model. By conducting experiments on a recently established benchmark Twitter dataset, aiming to address these research questions:

- **RQ1:** Can a multi-dialectal pre-trained language model, based on Twitter data, improve hate speech detection accuracy in Arabic tweets?
- **RQ2:** Does fine-tuning a pre-trained language model enhance hate speech detection accuracy in Arabic tweets?

First, we present the pre-processing step we have applied to the chosen datasets as well as the hyperparameters used for DL architectures and transfer learning models. Then, we discuss the evaluation metrics and finally present the achieved performance results.

3.1 Tweet Preprocessing

The preprocessing stage plays a pivotal role in natural language processing systems, particularly

in text classification [42]. They contain elongated words, hashtags, user mentions, and expressions that make tokenization difficult. To mitigate these challenges, we implement the following steps:

- Removing tweet features: This involves eliminating user URLs, mentions '@', hashtag symbols '#', punctuation, the word "RT", special characters (emoticons), and numerical characters.
- Removing non-Arabic letters, Arabic stop words, diacritics and new lines.
- Eliminating repeated characters: like (مرحباااااا) which means "Hellooooooo", to be (مرحبا), which is "Hello".
- Arabic letters standardization:
  - The letter (Taa Marbouta) (ة), which can be mistaken and written as (ه), we standardize it to (ه).
  - The Letter (Alef) (أ), which has the following forms (ا-آ-إ-أ), all the four letters were standardized into (ا).
  - The Arabic dash that is used for expanding words like in (مرحبـــا) to be (مرحبا).
  - The Letter (Alef Maqsora) (ى) has been standardized to (ي).

3.2 Hyperparameter Optimization for DL Models

The DL architectures evaluated in this work contain numerous hyperparameter, which necessitate estimation to achieve optimal results. To achieve this, we used the performance of a validation dataset to select the most suitable hyperparameter for the test dataset. For the hyperparameter optimization, we conduct 10-folds cross-validation using the corresponding dataset.

We employ the test data to make predictions while evaluating the predictions based on the optimized hyperparameters.

Table 2 illustrates the optimal hyperparameter for the corresponding model (LSTM, CNN, and BiLSTM). To avoid over-fitting during the supervised training of a neural network, we utilize early stopping by ending the training procedure before the converging of the weights.

Table 6. F1 score results of pre-trained language models after fine-tuning

Model	Macro	Weighted
xlm-roberta-base	52.57	74.57
xlm-roberta-large	48.98	72.74
bert-base-arabic	56.43	76.71
bert-large-arabic	55.30	75.68
bert-base-arabert	56.59	76.89
bert-large-arabert	57.05	76.65
base-multilingual-cased	48.91	73.26
multi-dialect-bert-base-arabic	65.95	81.86
albert-base-arabic	61.26	79.20
albert-large-arabic	62.21	79.74
base-arabertv02-twitter	**70.76**	**84.69**
large-arabertv02-twitter	69.71	84.45

3.3 Transfer Learning Fine-Tuning

The transformers-based models used in this work are pre-trained trained based on formal general corpora (Arabert, XLM-RoBERTa, and Multilanguage-BERT) and based on informal corpora (i.e. AraBERT-Twitter).

Thus, it is important to study the contextual information derived from the pre-trained layers while fine-tuning it for our interested downstream task. The fine-tuning consists of updating weights using the annotated dataset. BERT takes a sequence of 512 tokens as input and outputs 12 self-attention heads and a 768-dimensional vector.

For the optimization, we use the Adam optimizer [44,45] which performs well for natural language processing and the BERT model specifically. Additionaly, we evaluate other multilingual model, we chose the xlm-roberta-base and xlm-roberta-large checkpoints which include 100 languages.

For the purposes of fine-tuning, authors in [46] have recommended choosing from the values of the following parameters: number of epochs, batch size, learning rate, and maximum sequence. We fine-tuned the corresponding models by evaluating different parameters as presented in Table 3. We

Table 7. Comparison of F1 score results with the latest state-of-the-art classification approaches (N=Non-hate speech, S=Sexism, Re=Religious, Ra=Racial, M=Macro-averaged)

	N	S	Re	G	Ra	M
A	0.85	0.21	0.10	0.12	0.09	0.27
B	0.86	0.42	0.50	0.25	0.24	0.45
C	0.87	0.41	0.54	0.30	0.29	0.48
D	0.86	0.43	0.59	0.31	0.25	0.49
E	0.85	0.42	0.56	0.45	0.20	0.50
F	0.87	0.50	0.64	0.49	0.27	0.55
G	0.86	0.44	0.59	0.47	0.22	0.52
H	0.87	0.37	0.47	0.28	0.26	0.45
I	0.87	0.49	0.56	0.47	0.24	0.53
J	0.88	0.49	0.67	0.41	0.37	0.57
K	0.89	0.50	0.73	0.42	0.32	0.57
O	**0.92**	**0.70**	**0.80**	**0.61**	**0.50**	**0.71**

set the maximum sequence length as 64 for all the experiments.

3.4 Evaluation Metrics

To assess the effectiveness of the corresponding approaches, we will use various evaluation measures capable of accurately assessing the model's performance. Given that hate speech classification poses the imbalanced learning challenge, we will particularly emphasize the Macro-average and Weighted-average metrics to compute comprehensive performance metrics.

These metrics are presented as follow: Precision (P), referred to as positive predictive value, indicates the proportion of correctly classified positive instances out of all instances classified as positive. For example, the Precision of the Normal class is estimated as follows:

$$P_{Normal} = CC_{Normal} / TC_{Normal}, \quad (1)$$

where CC_{Normal} is the Correctly Classified as Normal and TC_{Normal} is the Total Classified as Normal. Recall (R) (also known as sensitivity), is the division of correctly predicted positive instances to the total positive instances. For example, the Recall of the Normal class is calculated as follows:

$$R_{Normal} = CC_{Normal} / TN_{Normal}, \quad (2)$$

where CC_{Normal} is the number of Correctly Classified as Normal, and TN_{Normal} is the Total number of Normal instances. F1 measure (F1) is the harmonic mean between the Precision and Recall. For example, the F1 of the Normal class is estimated as follows:

$$F1_{Normal} = 2(P_{Normal} \times R_{Normal}) / (P_{Normal} + R_{Normal}). \quad (3)$$

3.5 Results Analysis

In this subsection, we explore the achieved results of the approaches we evaluated. We employed the tenfold cross-validation method to evaluate performance metrics. This method involved splitting the dataset into 10 equally sized parts while maintaining a balanced representation of each class from the original dataset. One part was designated for testing, while the remaining parts were utilized for training.

This process was repeated 10 times, and the performance metric scores were averaged across the 10 iterations of cross-validation. Table 4 presents the prediction performances attained for both conventional machines leaning and DL classifiers. We utilized word embedding as an input feature vector in all models.

In the case of traditional machine learning approaches, also we tested also the statistical-features BoW and TF–IDF including SVM-BoW, SVM-TF-IDF, MNB-BoW, and MNB-TF–IDF. In most traditional classifiers, BoW achieves better performance results than the TF-IDF and word embedding features.

The baseline experiments (DL and traditional learning approaches) did not perform satisfactorily due to an insufficient number of training instances. The DL-based approaches achieve better performance results than conventional machine learning approaches.

These findings are in line with the majority of related works, where DL based approaches found to have comparable accuracy results to traditional learning algorithms on the corresponding task.

Table 8. The number of misclassified tweets by best pre-trained language model. (TE = Total Error, Q = All in Common)

	N	S	Re	G	Ra	TE
X	604	201	162	571	247	1785
Y	631	240	185	535	203	1794
W	890	183	166	536	176	1951
Z	713	240	160	545	243	1865
Q	334	45	30	160	57	726

As highlighted in Table 5, the pre-trained language model before fine-tuning substantially outperforms the baseline systems. It is important to note that multidialectal-based transformer models achieve better results than monolingual and multilingual-based models, yielding F1 score results of 63.38% and 80.85% for Macro averaged and Weighted averaged, respectively.

Table 6 shows the accuracy results of the pre-trained language model after fine-tuning. It is important to note that the fine-tuning of the pre-trained language model improves the accuracy results. Additionally, the multidialectal-based models outperform monolingual and multilingual ones, yielded F1 score results of 70.76% and 84.69% for Macro averaged and Weighted averaged, respectively.

Subsequently, we compare the performance of fine-tuned multidialectal-based model base-arabertv02-twitter (O) with three existing conventional machine learning approaches:

a. [12], which leveraged N-grams with NB.
b. [11], which combine TF–IDF and BoW with Bagging.
c. [8], which use TF-IDF with SVM, and seven existing DL-based approaches, namely.
d. [14], that learned word Embedding using the training data and use the hybrid CNN-LSTM for classification.
e. [19], Multilingual BERT embedding model with CNN.
f. [20], which use AraVec and bidirectional GRU augmented with attention layer.
g. [17], Which used AraBERT and AraVec embedding with CNN.
h. [15], which used AraVec and hybrid CNN-LSTM.
i. [18], which used AraBERT embedding and ensemble CNNs.
j. [21], which fine-tuned the pre-trained AraBERT language model.
k. [47], which use base-arabertv02-twitter without fine tuning. When comparing our approach with other state-of-the-art classifiers presented in Table 7, our model exhibits the highest Accuracy.

Unlike CNN and LSTM, our method does not necessitate a substantial quantity of labeled dataset to achieve promising performance result; this is a common requirement in many DL approaches. Moreover, in contrast to NB and SVM, our model eliminates the need to extract and design handcrafted features.

Given the nature of social media data, characterized by frequent usage of slang, abbreviations, and informal language, our method effectively processes the input words while considering their contextual surroundings.

Experimental results demonstrated that our proposition outperforms existing approaches by a difference between [8% and 21%] and [14% and 44 %] in weighted averaged and macro averaged F1 scores respectively. Comparing these results, we highlight the significance of multidialectal-based models trained on Twitter data since those models achieve the best results.

Furthermore, we highlight the significance of parameter tuning to discover the optimal hyperparameter values. Subsequently, we conduct an error analysis on the best pre-trained language models. For each model, we scrutinize tweets that were misclassified.

Furthermore, we examine tweets that were misclassified by all four models. Table 8 showed the number of tweets misclassified by such pre-trained language model.

The four best accurate models (bert-base-arabertv02-twitter (X), bert-large-arabertv02-twitter (Y), bert-large-arabic (W), and bert-base-arabert (Z)) predicted the same wrong labels 726 times out of 5217.Regarding the best system bert-base-

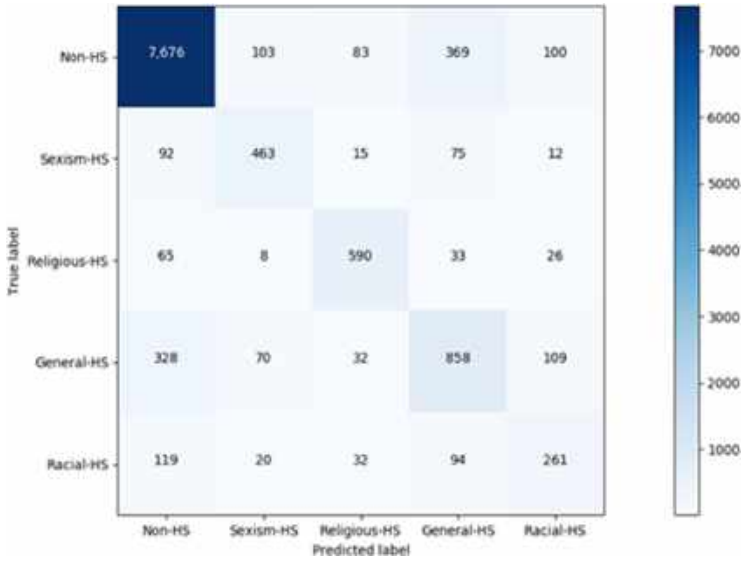


Fig. 1. Confusion matrix of our proposition

arabertv02-twitter, the Non-HS instances mislabeled are biased towards General-HS; the Sexism-HS, Religious-HS, Racial-HS, and General-HS instances mislabeled are biased towards Non-HS. Fig. 1 shows the confusion matrix of the optimized system.

4 Discussion

Although quite effective, current Arabic hate speech classification approaches are costly, as they need a huge number of labeled tweets to attain promising accuracy results. The tweets labeling process is very expensive and labor-intensive while hindering the deployment of artificial intelligence systems in the industry.

In contrast, our proposition does not require a huge number of labeled datasets. Furthermore, the machine-learning approaches use hand-crafted features, which have confronted data sparseness and the curse of dimensionality. Conversely, ours automatically learn features from the textual data.

As can be shown in Table 4, the minor performance results are obtained using the traditional learning classifiers followed by the deep learning classifiers, while the major accuracy results go for transfer learning-based classifiers. In Tables 5 and 6, we can notice that the fine-tuning of pre-trained language models improves the accuracy of results.

Furthermore, the multidialectal pre-trained language models based on Twitter data outperform monolingual and multilingual ones. In Table 7, we can notice that our proposition outperformed the latest state-of-the-art. The lower performance results are observed for the racial class, and the higher performance are obtained for the normal class. This disparity can be attributed to the significant class label imbalance present in the dataset.

6 Conclusion

Today, the detection of hate speech from Arabic tweets has garnered significant attention from scholars worldwide.

In this paper, we evaluate Arabic hate speech classification by utilizing transfer learning based on a pre-trained language model. We conducted extensive experiments following three approaches: two conventional machine-learning approaches, three DL approaches, and twelve transfer-learning approaches.

The results achieved by the transfer learning approaches outperform traditional and deep learning models utilized in this work. The major

contribution of this work is the evaluation of the recent pre-trained language models for Arabic hate speech classification. Specifically, we differentiate the performance of multilingual models with monolingual and multidialectal ones.

Experimental results show that the multidialectal models trained on Twitter data outperformed monolingual and multilingual models trained on general data. In our future work, we intend to pursue various avenues. Primarily, we aim to refine the contextual embedding model, with a focus on adapting its vocabulary for the hate speech classification task. A costlier technique could be to consider training a novel AraBERT model that is customized for Arabic hate speech classification.

Additionally, we intend to evaluate various data augmentation approaches to overcome the challenges of imbalanced data. From a research standpoint, we will utilize our proposed systems to examine Arabic Twitter discussions on various subjects to determine the extent of hate speech conversations with public discourse and to understand how their capabilities and sophistication evolve.

References

1. **Council of Europe (2016).** ECRI general policy recommendation on combating hate speech. European Commission against Racism and Intolerance General Policy Recommendation, No. 15. hudoc.ecri.coe.int/eng?i=REC-15-2016-015-ENG.
2. **Hinduja, S., Patchin, J. W. (2010).** Bullying, cyberbullying, and suicide. Archives of Suicide Research, Vol. 14, No. 3, pp. 206–221. DOI: 10.1080/13811118.2010.494133.
3. **Fortuna, P., Nunes, S. (2018).** A survey on automatic detection of hate speech in text. ACM Computing Surveys, Vol. 51, No. 4, pp. 1–30. DOI: 10.1145/3232676.
4. **Daouadi, K. E., Rebaï, R. Z., Amous, I. (2021).** Optimizing semantic deep forest for tweet topic classification. Information Systems, Vol. 101, pp. 101801. DOI: 10.1016/j.is.2021.101801.
5. **Daouadi, K. E., Rebai, R. Z., Amous, I. (2018).** Organization vs. individual: Twitter user classification. Conference on Language Processing and Knowledge Management, pp. 1–8.
6. **Daouadi, K. E., Rebaï, R. Z., Amous, I. (2019).** Organization, bot, or human: towards an efficient twitter user classification. Computación y Sistemas, Vol. 23, No. 2, pp. 273–279. DOI: 10.13053/cys-23-2-3192.
7. **Daouadi, K. E., Rebaï, R. Z., Amous, I. (2019).** Bot detection on online social networks using deep forest. Artificial Intelligence Methods in Intelligent Algorithms, Vol. 985, pp. 307–315. DOI: 10.1007/978-3-030-19810-7_30.
8. **Chowdhury, S. A., Mubarak, H., Abdelali, A., Jung, S., Jansen, B. J., Salminen, J. (2020).** A multi-platform arabic news comment dataset for offensive language detection. Proceedings of the 12th Language Resources and Evaluation Conference, pp. 6203–6212.
9. **Husain, F. (2020).** OSACT4 shared task on offensive language detection: Intensive preprocessing-based approach. Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, pp. 53–60.
10. **Daouadi, K. E., Boualleg, Y., Guehairia, O. (2023).** Deep random forest and araBert for hate speech detection from Arabic tweets. JUCS - Journal of Universal Computer Science, Vol. 29, No. 11, pp. 1319–1335. DOI: 10.3897/jucs.112604.
11. **Husain, F. (2020).** Arabic offensive language detection using machine learning and ensemble machine learning approaches. DOI: 10.48550/ARXIV.2005.08946.
12. **Mulki, H., Haddad, H., Ali, C. B., Alshabani, H. (2019).** L-HSAB: A Levantine twitter dataset for hate speech and abusive language. Proceedings of the Third Workshop on abusive Language Online. DOI: 10.18653/v1/w19-3512.
13. **Abozinadah, E. A., Jones, J. H. (2017).** A statistical learning approach to detect abusive twitter accounts. Proceedings of the

International Conference on Compute and Data Analysis, pp. 6–13. DOI: 10.1145/3093241.3093281.

14. **Al-Hassan, A., Al-Dossari, H. (2021).** Detection of hate speech in Arabic tweets using deep learning. Multimedia Systems, Vol. 28, No. 6, pp. 1963–1974. DOI: 10.1007/s00530-020-00742-w.
15. **Faris, H., Aljarah, I., Habib, M., Castillo, P. (2020).** Hate speech detection using word embedding and deep learning in the Arabic language context. Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods, pp. 453–460. DOI: 10.5220/0008954004530460.
16. **Alsafari, S., Sadaoui, S., Mouhoub, M. (2020).** Effect of word embedding models on hate and offensive speech detection. arXiv. DOI: 10.48550/ARXIV.2012.07534.
17. **Alghanmi, I., Anke, L. E., Schockaert, S. (2020).** Combining BERT with static word embeddings for categorizing social media. Proceedings of the 6th Workshop on Noisy User-generated Text. DOI: 10.18653/v1/2020.wnut-1.5.
18. **Alsafari, S., Sadaoui, S., Mouhoub, M. (2020).** Deep learning ensembles for hate speech detection. IEEE 32nd International Conference on Tools with Artificial Intelligence, pp. 526–531. DOI: 10.1109/ictai50040.2020.00087.
19. **Alsafari, S., Sadaoui, S., Mouhoub, M. (2020).** Hate and offensive speech detection on Arabic social media. Online Social Networks and Media, Vol. 19, pp. 100096. DOI: 10.1016 /j.osnem.2020.100096.
20. **Haddad, B., Orabe, Z., Al-Abood, A., Ghneim, N. (2020).** Arabic offensive language detection with attention-based deep neural networks. Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, pp. 76–81.
21. **Mubarak, H., Rashed, A., Darwish, K., Samih, Y., Abdelali, A. (2021).** Arabic offensive language on twitter: analysis and experiments. Proceedings of the 6th Arabic Natural Language Processing Workshop, pp. 126–135.
22. **De-Sousa-Pereira, A. B., Firmino-Alves, A. L., De-Oliveira, M. G., Baptista, C. D. (2018).** Using supervised classification to detect political tweets with political content. Proceedings of the 24th Brazilian Symposium on Multimedia and the Web, pp. 245–252. DOI: 10.1145/3243082.3243113.
23. **Garreta, R., Moncecchi, G. (2013).** Learning scikit-learn: Machine learning in python. Packt Publishing Ltd.
24. **Yang, X., Macdonald, C., Ounis, I. (2017).** Using word embeddings in twitter election classification. Information Retrieval Journal, Vol. 21, No. 2-3, pp. 183–207. DOI: 10.1007/s10791-017-9319-5.
25. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J. (2013).** Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems, Vol. 26, pp. 3111–3119.
26. **Pennington, J., Socher, R., Manning, C. (2014).** Glove: global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1532–1543. DOI: 10.3115/v1/D14-1162.
27. **Joulin, A., Grave, E., Bojanowski, P., Mikolov, T. (2017).** Bag of tricks for efficient text classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Vol. 2, pp. 427–431.
28. **Devlin, J., Chang, M., Lee, K., Toutanova, K. (2019).** BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.
29. **Cortes, C., Vapnik, V. (1995).** Support-vector networks. Machine Learning, Vol. 20, No. 3, pp. 273–297. DOI: 10.1007/bf00994018.
30. **Ruiz, A. M., Cornet, A., Shimanoe, K., Morante, J. R., Yamazoe, N. (2005).** Effects

of various metal additives on the gas sensing performances of tio2 nanocrystals obtained from hydrothermal treatments. Sensors and Actuators B: Chemical, Vol. 108, No. 1-2, pp. 34–40. DOI: 10.1016/j.snb.2004.09.045.

31. **Mccallum, A., Nigam, K. (1998).** A comparison of event models for Naive Bayes text classification. AAAI Conference on Artificial Intelligence, pp. 41–48.
32. **Kim, Y. (2014).** Convolutional neural networks for sentence classification. Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1746–1751. DOI: 10.3115/v1/D14-1181.
33. **Mikolov, T., Karafiát, M., Burget, L., Černocký, J., Khudanpur, S. (2010).** Recurrent neural network based language model. Proceedings of the 11th Interspeech, pp. 1045–1048. DOI: 10.21437/Interspeech. 2010-343.
34. **Hochreiter, S., Schmidhuber, J. (1997).** Long short-term memory. Neural Computation, Vol. 9, No. 8, pp. 1735–1780. DOI: 10.1162/ neco.1997.9.8.1735.
35. **Schuster, M., Paliwal, K. (1997).** Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, Vol. 45, No. 11, pp. 2673–2681. DOI: 10.1109/78.650093.
36. **Soliman, A. B., Eissa, K., El-Beltagy, S. R. (2017).** AraVec: A set of Arabic word embedding models for use in Arabic NLP. Procedia Computer Science, Vol. 117, pp. 256–265. DOI: 10.1016/j.procs.2017.10.117.
37. **Torrey, L., Shavlik, J. (2010).** Transfer learning. Handbook of Research on Machine Learning Applications and Trends, pp. 242–264. DOI: 10.4018/978-1-60566-766-9.ch011.
38. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2017).** Attention is all you need. Advances in Neural Information Processing Systems, Vol. 30, pp. 5998–6008.
39. **Conneau, A., Lample, G. (2019).** Cross-lingual language model pretraining. Advances in Neural Information Processing Systems, Vol. 32, pp. 7057–7067.
40. **Gage, P. (1994).** A new algorithm for data compression. C Users Journal, Vol. 12, No. 2, pp. 23–38.
41. **Sennrich, R., Haddow, B., Birch, A. (2016).** Neural machine translation of rare words with subword units. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 1715–1725. DOI: 10.18653/v1/ P16-1162.
42. **Uysal, A. K., Gunal, S. (2014).** The impact of preprocessing on text classification. Information Processing and Management, Vol. 50, No. 1, pp. 104–112. DOI: 10.1016/j.ipm.20 13.08.006.
43. **Hutto, C., Gilbert, E. (2014).** Vader: a parsimonious rule-based model for sentiment analysis of social media text. Proceedings of the International AAAI Conference on Web and Social Media, Vol. 8, No. 1, pp. 216–225. DOI: 10.1609/icwsm.v8i1.14550.
44. **Kingma, D. P., Ba, J. (2014).** Adam: A method for stochastic optimization. 3rd International Conference for Learning Representations, pp. 1–15. DOI: 10.48550/ARXIV.1412.6980.
45. **Zhang, J., Karimireddy, S. P., Veit, A., Kim, S., Reddi, S. J., Kumar, S., Sra, S. (2020).** Why ADAM beats SGD for attention models. Proceedings of the International Conference on Learning Representations, pp. 1–18.
46. **Sun, C., Qiu, X., Xu, Y., Huang, X. (2019).** How to fine-tune Bert for text classification? Chinese Computational Linguistics, pp. 194–206. DOI: 10.1007/978-3-030-32381-3_16.
47. **Antoun, W., Baly, F., Hajj, H. (2020).** AraBERT: Transformer-based model for Arabic language understanding. Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, pp. 9– 15.

Article received on 26/01/2022; accepted on 24/04/2024.
**Corresponding author is Kheir Eddine Daouadi.*

ISSN 2007-9737

Quantum Classifier for Natural Language Processing Applications

Shyambabu Pandey[1], Partha Pakray[1,*], Riyanka Manna[2]

[1] National Institute of Technology Silchar, Assam, Silchar, India

[2] Amrita Vishwa Vidyapeetham, Amaravati, Ettimadai, India

{babushyampandey, parthapakray, riyankamanna16}@gmail.com

Abstract. A deep neural network is a branch of machine learning that is capable of learning and representing complex patterns from a dataset through interconnected multiple layers of neurons. This capability makes it applicable in various fields, such as natural language processing, image processing, and computer vision. Deep learning models show effective performance but face challenges such as complexity and resource demands. On the other hand, quantum machine learning algorithms offer an alternative with potential efficiency compared to their classical counterparts. This paper proposes a Quantum Recurrent Neural Network (QRNN) for natural language processing tasks, which classify text data such as parts of speech, named entity recognition, and sentiment analysis. The proposed method utilizes parameterized quantum circuits that contain the tunable parameters. Our approach uses amplitude encoding to represent classical data into quantum states, partial measurement for label determination, and ancilla qubits to transfer the information from the current state to the next.

Keywords. Quantum computing, quantum machine learning, natural language processing.

1 Introduction

Machine learning [1] is a branch of artificial intelligence [22] that allows computers to learn patterns in a dataset without using explicit programs. The main goal of machine learning algorithms is to be applied in engineering and science fields, with the primary objective of identifying patterns and making decisions according to those patterns. Introducing deep neural networks [23] enhances the performance and capability of machine learning algorithms, which can find more complex patterns from datasets.

Over the past few years, machine learning algorithms have applied deep learning models to enhance the performance of natural language processing (NLP) applications [24]. Specifically, transformer-based models perform better than traditional neural network models, such as recurrent neural network variants.

However, this superior performance poses different challenges, such as complexity in the model, demands extensive dataset, and raises a considerable amount of computational power, time, and resources [10].

An alternate path is obtaining notice in recent times from quantum computing, which can perform efficient computation for specific problems. Quantum computing [11] is a computing paradigm that follows the law of quantum theory and applies qubits (quantum bits) as processing components.

Qubit is analogous to classical bits capable of exciting in the superposition of all the possible states. Additionally, entanglement, a quantum principle, can exploit the correlation between multiple qubits, even if they are physically separated. Such unique characteristics of quantum phenomena make quantum computing solve specific problems faster than their classical solutions. This hope allows us to solve some computational issues efficiently, such as machine learning algorithms.

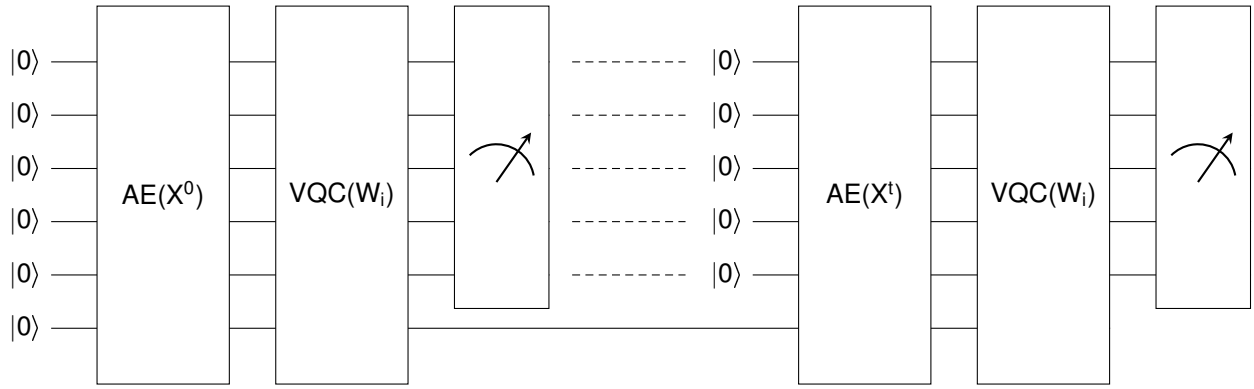


Fig. 1. Architecture of the proposed system

Quantum machine learning (QML) [18], a combination of quantum computing and machine learning to utilize powerful features taken from quantum theory to enhance the computational power of traditional machine learning algorithms. Some quantum machine learning algorithms show remarkable improvement compared to their classical counterparts.

Quantum neural network (QNN) [12] is a subfield of QML algorithms with more learning capabilities than classical neural networks. In recent times, researchers have endeavored to apply a quantum version of classical machine learning algorithms to NLP applications to improve performance.

Most NLP applications classify text data into different labels, such as part-of-speech (POS) tagging, named-entity recognition (NER), and sentimental analysis (SA). To enhance the performance of such NLP tasks, we propose a quantum recurrent neural network (QRNN) as a quantum classifier for the text data.

The proposed system is based on a parameterized quantum circuit consisting of tunable parameters to train the model and employ amplitude embedding to convert each word's word embedding into quantum states.

2 Related Work

Some QML algorithms show exponential speedup compared to their classical counterparts, such as learning algorithms like quantum principle component analysis [15], and quantum support vector machine [21]. However, these algorithms cannot run on current quantum computers because of the lack of quantum RAM to execute.

To take advantage of current quantum hardware, variational quantum circuits [5] are employed, which consist of a series of parameterized quantum gates. Quantum approximation optimization algorithm [8], hybrid quantum-classical algorithms [3], and QNN are examples of variational quantum algorithms. These algorithms are implemented on current quantum hardware systems, which use parameterized quantum circuits in short-depth.

Researchers are trying to implement hybrid quantum-classical QNN models using quantum computing principles and classical neural network architectures. For example, Liu et al. [14] have introduced a quantum convolutional neural network (QCNN) based on hybrid quantum-classical methodology.

Ceschini et al. [4] proposed a hybrid quantum-classical recurrent neural network to predict time series data of renewable energy. Chen et al. [6] introduced a hybrid model for quantum long short-term memory (QLSTM) that can apply to NISQ devices to handle sequential data. QNN models can applied in NLP applications to enhance their performance.

Sipio et al. [7] introduced a QLSTM hybrid model and employed it to perform POS tagging. This model demonstrates an attempt to employ

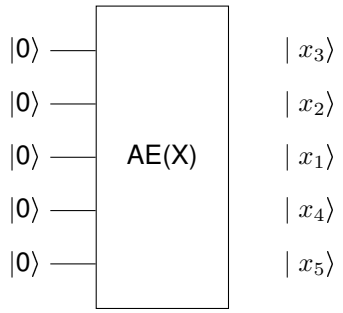


Fig. 2. Circuit for data encoding. Initially, all quantum states are present in default state $|0\rangle$

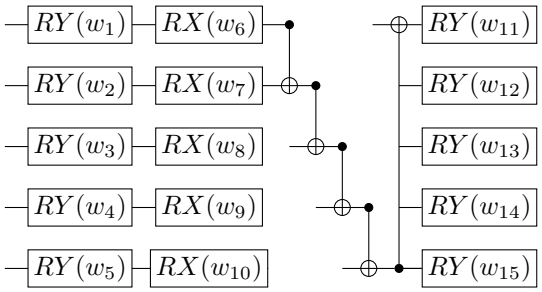


Fig. 3. Circuit for VQC

NLP applications by using QNN. However, this model does not show any significant advantage for POS tagging. Pandey et al. [19] implements a quantum LSTM (QLSTM) for POS tagging of a low-resource Indian language, Mizo.

The authors experimented with different numbers of qubits and performed hyperparameter tuning. However, their experiment's result could be better, showing that current quantum devices do not apply to large datasets. Pandey et al. [18] propose a hybrid quantum-classical QLSTM for POS tagging on code-mixed languages.

This model converts gates of classical LSTM into variational quantum layers. However, the proposed is not able to process large datasets. So, the authors split entire datasets into batches of hundred sentences for the experiments.

The code-mixed dataset consists of nine datasets collected from three social media platforms: Facebook, Twitter, and WhatsApp. The authors [13] propose QRNN to handle sequential data.

The proposed model is employed to predict stock prices and classify the text data that show significant improvements. However, this model applies to small datasets. Quantum natural language processing (QNLP) [16] is another research area to utilize near-term quantum computers for NLP applications. It employs compositional distributional semantics (DisCoCat) that apply the compositional structure of the Pregroup grammars.

It represents the grammatical structure of sentences as a string diagram, encoding a specific interaction of words according to the grammar. The DiscoCat converts these string diagrams into quantum circuits to process NLP applications. Various NLP applications have been implemented by the QNLP framework, like question answering [17], grammar-aware classification [17], and sentiment analysis [9]. QNLP demands a massive amount of computational resources, which makes it time-consuming to process NLP applications.

3 System Architecture

We propose a QRNN that presents a quantum counterpart of classical RNN architecture. The QRNN employs variational quantum circuits (VQC) consisting of parameterized quantum gates. These gates imply tunable parameters, which provide flexibility during the model training. In QRNN, each cell of traditional RNN is substituted by VQC. Our main objective is to perform the classification of text into different labels.

So, we perform a measurement of each circuit that provides labels for each input. The structure of the QRNN is divided into three submodules: data encoding, VQC, and measurement. Figure 1 represents the architecture of the proposed system, where AE represents amplitude encoding, and VQC represents a variational quantum circuit.

The given circuit is quantum analogous to classical RNN, where each VQC represent the cells of RNNs. Each time instance, data X_t must be encoded and applied to a VQC to process the data. In the end, partial measurements are employed to determine the labels of corresponding text data.

Again, we initialize the quantum circuit as the initial state and pass the next data instance. We use one ancilla qubit that passes the previous information to the current state, which serves as the hidden state of RNN.

3.1 Encoding Method

We must convert classical information to quantum states to process classical data using the quantum framework. The process of transforming classical data into quantum state is known as data encoding. Various data encoding methods exist, such as basis, amplitude, and angle encoding. In our study, we use amplitude encoding due to its compatibility with our quantum classifier.

Amplitude encoding encodes classical data into the amplitudes of quantum states. It uses the principle of superposition and allows multiple information to be represented in a single quantum state. The choice of amplitude encoding is because it requires fewer qubits to describe the sizeable dimensional dataset. Suppose an N qubit employs amplitude encoding; then it holds 2^n quantum states. Moreover, our systems mainly focus on NLP applications, which always deal with large dimensional datasets.

State-of-the-art (SOTA) QNN models utilize angle encoding for data encoding, where each qubit generates a single quantum state. Handling extensive dimensional data requires more qubits, while current quantum hardware limitations prevent handling large numbers of qubits. As a result, SOTA QNNS can not manage large datasets. Meanwhile, word embedding of each word generates high-dimensional word vectors.

So, our novel approach uses amplitude encoding in QRNN to handle large dimensional word vectors.

3.2 Variational Quantum Circuit

After encoding classical information, the next step applies VQCs to process quantum states. Figure 3 represents a VQC of the proposed system, which consists of Y and X rotational and controlled-not gates. The processing unit of the neural network is a combination of linear and non-linear operations.

So, our proposed system contains rotational gates X and Y, which have parameterized gates consisting of adjustable parameters, where the Y gate can represent non-linear operation [2]. The X gate is a cable that represents linear operations. A controlled-not gate is applied to generate entanglement between different qubits, which increases the circuit's entangling capability.

It helps to identify the patterns between data. The parameters $w_1,\ w_2,\ ...,\ w_{15}$ are adjustable to train the model. Our goal is to use fewer parameters to learn the model, thereby reducing the overload of many parameters during training. Classical deep learning models demonstrate good performances but require a substantial number of parameters to train the models, which makes them complex models.

On the other hand, quantum computers can find patterns from fewer parameters [20]. This unique property of quantum computers makes them applicable for identifying the patterns in datasets. However, current quantum hardware is limited in handling many parameters. Therefore, our proposed model uses fewer parameters for training, enabling it to be compatible with existing quantum hardware.

3.3 Measurement

Finally, the circuit is measured, generating classical information that will give a label of an observation. The proposed uses partial measurement, which measures all the quantum states except the ancilla qubit. The use of ancilla qubit passes the previous information to the current state.

Tila qubits serve as a hidden state of classical RNN, which maintains the flow of information from the prior state to the current state. The result of the measurement assigns labels of corresponding words. Various measurement methods exist to measure the quantum circuits. However, we use expectation measurement that provides the expected value of each observation.

4 Conclusion & Future Work

We introduce a QRNN as a quantum classifier to perform classification tasks on text data. The proposed system uses quantum mechanics principles to enhance the performance of NLP tasks. The present architecture is a novel approach, applying amplitude encoding to encode classical information and employing partial measurement to determine the label of text data, with an ancilla qubit that passes the previous state information to the current state.

We designed our proposed QRNN to accommodate sizeable dimensional word vectors, maintaining each word's integrity and requiring fewer parameters to train the model, making it compatible with current quantum computers. Our future work will apply the proposed model as a quantum classifier to classify texts such as POS tagging, NER, and text classification.

Acknowledgments

The work presented here falls under Project Entitled "Quantum Machine Learning for Natural Language Processing Applications" (Grant Ref. No. N-21/17/2020-NeGD) supported by MeitY Quantum Computing Applications Lab (QCAL).

References

1. **Baştanlar, Y., Özuysal, M. (2014).** Introduction to machine learning. miRNomics: MicroRNA biology and computational analysis, Vol. 1107, pp. 105–128. DOI: 10.1007/978-1-62703-748-8_7.
2. **Bausch, J. (2020).** Recurrent quantum neural networks. Proceedings of the 34th Conference on Neural Information Processing Systems, pp. 1368–1379. DOI: 10.48550/ARXIV.2006.14619.
3. **Callison, A., Chancellor, N. (2022).** Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond. Physical Review A, Vol. 106, No. 1. DOI: 10.1103/PhysRevA.106.010101.
4. **Ceschini, A., Rosato, A., Panella, M. (2022).** Hybrid quantum-classical recurrent neural networks for time series prediction. Proceedings of the International Joint Conference on Neural Networks, pp. 1–8. DOI: 10.1109/IJCNN55064.2022.9892441.
5. **Chen, S. Y. C., Huck-Yang, C. H., Qi, J., Chen, P. Y., Ma, X., Goan, H. S. (2020).** Variational quantum circuits for deep reinforcement learning. IEEE Access, Vol. 8, pp. 141007–141024. DOI: 10.1109/ACCESS.2020.3010470.
6. **Chen, S. Y. C., Yoo, S., Fang, Y. L. L. (2022).** Quantum long short-term memory. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8622–8626. DOI: 10.1109/icassp43922.2022.9747369.
7. **Di-Sipio, R., Huang, J. H., Chen, S. Y. C., Mangini, S., Worring, M. (2022).** The dawn of quantum natural language processing. IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8612–8616. DOI: 10.1109/icassp43922.2022.9747675.
8. **Farhi, E., Goldstone, J., Gutmann, S. (2014).** A quantum approximate optimization algorithm. arXiv. DOI: 10.48550/ARXIV.1411.4028.
9. **Ganguly, S., Morapakula, S. N., Coronado, L. M. P. (2022).** Quantum natural language processing based sentiment analysis using lambeq toolkit. Proceedings of the 2nd International Conference on Power, Control and Computing Technologies, pp. 1–6. DOI: 10.1109/ICPC2T53885.2022.9776836.
10. **Guarasci, R., De-Pietro, G., Esposito, M. (2022).** Quantum natural language processing: Challenges and opportunities. Applied Sciences, Vol. 12, No. 11, pp. 5651. DOI: 10.3390/app12115651.
11. **Gyongyosi, L., Imre, S. (2019).** A survey on quantum computing technology. Computer Science Review, Vol. 31, pp. 51–71. DOI: 10.1016/j.cosrev.2018.11.002.

12. **Jia, Z. A., Yi, B., Zhai, R., Wu, Y. C., Guo, G. C., Guo, G. P. (2019).** Quantum neural network states: A brief review of methods and applications. Advanced Quantum Technologies, Vol. 2, No. 7–8. DOI: 10.1002/qute.201800077.

13. **Li, Y., Wang, Z., Han, R., Shi, S., Li, J., Shang, R., Zheng, H., Zhong, G., Gu, Y. (2023).** Quantum recurrent neural networks for sequential learning. Neural Networks, Vol. 166, pp. 148–161. DOI: 10.1016/j.neunet.2023.07.003.

14. **Liu, J., Lim, K. H., Wood, K. L., Huang, W., Guo, C., Huang, H. L. (2021).** Hybrid quantum-classical convolutional neural networks. Science China Physics, Mechanics and Astronomy, Vol. 64, No. 9, pp. 1–8. DOI: 10.1007/s11433-021-1734-3.

15. **Lloyd, S., Mohseni, M., Rebentrost, P. (2014).** Quantum principal component analysis. Nature Physics, Vol. 10, No. 9, pp. 631–633. DOI: 10.1038/nphys3029.

16. **Meichanetzidis, K., Gogioso, S., de-Felice, G., Chiappori, N., Toumi, A., Coecke, B. (2021).** Quantum natural language processing on near-term quantum computers. Electronic Proceedings in Theoretical Computer Science, Vol. 340, pp. 213–229. DOI: 10.4204/eptcs.340.11.

17. **Meichanetzidis, K., Toumi, A., de-Felice, G., Coecke, B. (2023).** Grammar-aware sentence classification on quantum computers. Quantum Machine Intelligence, Vol. 5, No. 1. DOI: 10.1007/s42484-023-00097-1.

18. **Pandey, S., Basisth, N. J., Sachan, T., Kumari, N., Pakray, P. (2023).** Quantum machine learning for natural language processing application. Physica A: Statistical Mechanics and its Applications, Vol. 627, pp. 129123. DOI: 10.1016/j.physa.2023.129123.

19. **Pandey, S., Dadure, P., Nunsanga, M. V. L., Pakray, P. (2022).** Parts of speech tagging towards classical to quantum computing. Proceedings of the IEEE Silchar Subsection Conference, pp. 1–6. DOI: 10.1109/silcon55242.2022.10028796.

20. **Phillipson, F. (2020).** Quantum machine learning: Benefits and practical examples. Vol. 2561, pp. 51–56.

21. **Rebentrost, P., Mohseni, M., Lloyd, S. (2014).** Quantum support vector machine for big data classification. Physical review letters, Vol. 113, No. 13, pp. 130503. DOI: 10.1103/physrevlett.113.130503.

22. **Russell, S., Russell, S. J., Norvig, P., Davis, E. (2010).** Artificial intelligence: A modern approach. Prentice Hall.

23. **Suk, H. I. (2017).** An introduction to neural networks and deep learning. In Deep Learning for Medical Image Analysis. Elsevier, pp. 3–24.

24. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I. (2017).** Attention is all you need. Advances in Neural Information Processing Systems 30, Vol. 30, pp. 1–11.

Article received on 02/02/2024; accepted on 14/04/2024.
**Corresponding author is Partha Pakray.*

ISSN 2007-9737

Analysis of Relationships between Co-Symmetric Dissimilarity Measures of Probability Distributions with Involutive Negations

Maria Elena Ensastegui-Ortega[1], Ildar Batyrshin[1,*], Alexander Gelbukh[1], Nailya Kubysheva[2]

[1] Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico

[2] Kazan (Volga Region) Federal University, Republic of Tatarstan, Russia

{batyr1, elena.ensastegui}@gmail.com

Abstract. Learning from data in almost any human activity is a very important task, usually using similarity or dissimilarity between data. Recently, it was shown the importance of considering the involution operation defined on the data domain which reflects a symmetry of data structures. This symmetry should be taken into account in data analysis. Co-symmetric similarity and dissimilarity measures defined over a set with involution play an important role in data analysis. In this paper, four dissimilarity functions over the set of probability distributions are created that meet the property of co-symmetry with respect to the involutive negation of distributions. Scatter graphs are generated from their respective dissimilarity matrices to compare the similarity between them. Additionally, the Pearson, Kendall, and Spearman correlation coefficients are calculated to numerically assess the relationship that exists. Subsequently, four dissimilarity functions are considered due to their higher correlation with those studied in this paper. They are divided into two groups, and an analysis is conducted to determine which are more correlated.

Keywords: Co-symmetry, correlation, dissimilarity, involution, probability distribution.

1 Introduction

Many similarity and dissimilarity measures are proposed for probability distributions [1,2]. Recently, an involutive negation of probability distributions [3] and measure of correlation between distributions were introduced [4, 5]. This correlation measure used co-symmetric distance between probability distributions based on involutive negation of probability distributions. Co-symmetric similarity and dissimilarity measures are important for applications because they take into account the symmetry of data related to involution operation [6]. In this paper, four new co-symmetric dissimilarity functions for probability distributions are created and compared with the other co-symmetric distances between probability distributions considered in [7].

In Section 1, a small outline of the theory used to support the results is given. In Section 2, four distances are used to create four dissimilarity functions that comply with the co-symmetry property. In section 3, they are compared with four other co-symmetric distances introduced in [7].

2 Preliminary Definitions

2.1 Negator and Negation of Probability Distributions

Let $X = (x_1, \dots, x_d)$ be a set of alternatives ordered in some way. A probability distribution over X is a sequence of non-negative numbers $P = (P_1, \dots, P_d)$ such that $\sum_{i=1}^{d} P_i = 1$. Here, for all $i = 1, \dots, d$, P_i is considered as a probability of x_i.

The first example of negation of probability distributions was introduced in [8]. In [9], the

Table 1. Original distances that were considered for this analysis.

Name	Distance
Soergel	$d_{sg} = \frac{\sum_{i=1}^{d} \lvert P_i - Q_i \rvert}{\sum_{i=1}^{d} \max(P_i, Q_i)}$
Sørensen	$d_{sor} = \frac{\sum_{i=1}^{d} \lvert P_i - Q_i \rvert}{\sum_{i=1}^{d} (P_i + Q_i)}$
Jaccard	$d_{Jac} = \frac{\sum_{i=1}^{d} (P_i - Q_i)^2}{\sum_{i=1}^{d} {P_i}^2 + \sum_{i=1}^{d} {Q_i}^2 - \sum_{i=1}^{d} P_i Q_i}$
Dice	$d_{Jac} = \frac{\sum_{i=1}^{d} (P_i - Q_i)^2}{\sum_{i=1}^{d} {P_i}^2 + \sum_{i=1}^{d} {Q_i}^2}$

Table 2. New distances created from the original distances and equation (1)

Distance	$\boldsymbol{d_{Name+\,Co-Pro}(P,Q)}$
Soergel	$d_{sg-Co-Pro} = \frac{\sum_{i=1}^{d} \lvert P_i - Q_i \rvert \ \sum_{i=1}^{d} \lvert N(P_i) - N(Q_i) \rvert}{\sum_{i=1}^{d} \max(P_i, Q_i) \sum_{i=1}^{d} \max(N(P_i), N(Q_i))}$
Sørensen	$d_{sor-Co-Pro} = \frac{\sum_{i=1}^{d} \lvert P_i - Q_i \rvert \sum_{i=1}^{d} \lvert N(P_i) - N(Q_i) \rvert}{\sum_{i=1}^{d} (P_i + Q_i) \sum_{i=1}^{d} (N(P_i) + N(Q_i))}$
Jaccard	$d_{Jac-Co-Pro} = \frac{\sum_{i=1}^{d} (P_i - Q_i)^2 \sum_{i=1}^{d} (N(P_i) - N(Q_i))^2}{\left(\sum_{i=1}^{d} {P_i}^2 + \sum_{i=1}^{d} {Q_i}^2 - \sum_{i=1}^{d} P_i Q_i\right)\left(\sum_{i=1}^{d} N(P_i)^2 + \sum_{i=1}^{d} N(Q_i)^2 - \sum_{i=1}^{d} N(P_i) N(Q_i)\right)}$
Dice	$d_{Dice-Co-Pro} = \frac{\sum_{i=1}^{d} (P_i - Q_i)^2 \ \sum_{i=1}^{d} \left(N(P_i) - N(Q_i)\right)^2}{\sum_{i=1}^{d} \left({P_i}^2 + {Q_i}^2\right) \sum_{i=1}^{d} (N(P_i)^2 + N(Q_i)^2)}$

general properties of negations of probability distributions and the class on linear negations of probability distributions are considered. In [2], it was introduced an involutive negation of probability distributions.

Relationships of negation with entropy of probability distributions are studied in [10]. Interpretation of probability distributions as fuzzy distribution sets and extension on probability distributions parametric negations of fuzzy sets is considered in [11, 12].

A negator N is a function that transforms point to point one probability distribution $P = (\mathrm{P}_1, \dots, \mathrm{P}_\mathrm{d})$ into another probability distribution $neg(P) = N(P_1), \dots N(P_n))$ called negation of P [9], such that for all $i, j = 1, \dots, n$, from $P_i \le P_j$ it follows $N(P_i) \ge \left(P_j\right)$.

A negation is called an involutive if $neg\big(neg(P)\big) = P$. In [3], Batyrshin introduced a negator:

$$N_B(P_i) = \frac{\max(P) + \min(P) - P_i}{n(\max(P) + \min(P)) - 1} = \frac{MP - P_i}{nMP - 1},$$

where $\max(P) = \max_{i=1,\dots,n}\{P_i\}$, $\min(P) = \min_{i=1,\dots,n}\{P_i\}$, $MP = \max(P) + \min(P)$. This negator generates an involutive negation of probability distributions: $neg_B(P) = \big(N_B(p_1), \dots, N_B(p_n)\big)$ such that $neg_B\big(neg_B(P)\big) = P$.

2.2 Co-symmetric Dissimilarity Functions

Suppose $P = (P_1, \dots, P_d)$ and $Q = (Q_1, \dots, Q_d)$ are two probability distributions. A dissimilarity function $D(P, Q)$ takes values in the interval [0,1] and satisfy the following properties:

symmetry:

$D(P, Q) = D(Q, P)$,

irreflexivity:

$D(P, P) = 0$.

Table 3. Pearson, Kendall and Spearman coefficients for distances creating from equation (1)

Distance	Pearson	Kendall	Spearman
Sorensen Co-Pro Vs Soergel Co-Pro	0,9919	0,9613	0,9976
Sorensen Co-Pro Vs Jaccard Co-Pro	0,9398	0,7828	0,9346
Sorensen Co-Pro Vs Dice Co-Pro	0,9310	0,7705	0,9268
Soergel Co-Pro Vs Jaccard Co-Pro	0,9372	0,7750	0,9299
Soergel Co-Pro Vs Dice Co-Pro	0,9137	0,7580	0,9187
Jaccard Co-Pro Vs Dice Co-Pro	0,9903	0,9634	0,9978

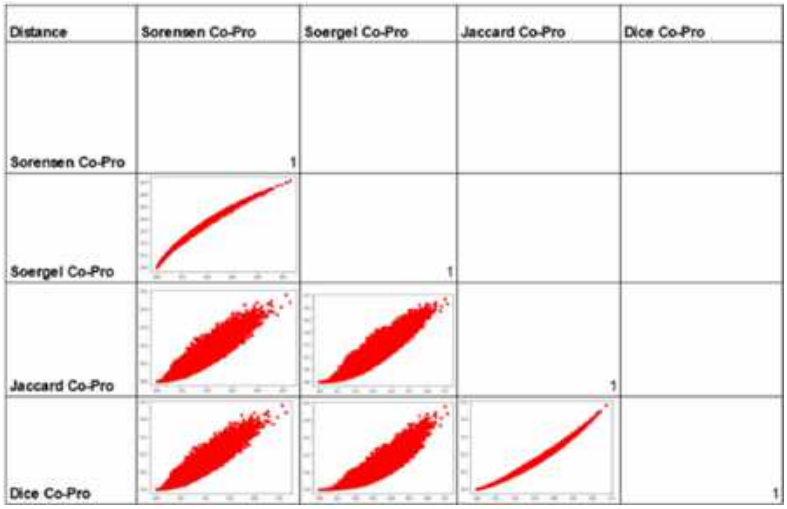


Fig. 1. Scatter graphs comparing distances created from equation (1)

Dissimilarity function is co-symmetric if for all probability distributions P and Q of the length n, it is fulfilled:

$$D\left(neg_B(P), neg_B(Q)\right) = D(P, Q).$$

2.3 Correlation Coefficients

Pearson's correlation coefficient, commonly used in statistical analyses, allows the evaluation of the presence and strength of a linear relationship between two quantitative variables. It varies between -1 and 1.

A value of 1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 suggests no linear correlation. On the other hand, Spearman and Kendall correlations are useful tools to investigate monotonic relationships between variables.

While Spearman's is based on the ranges of observations, Kendall's focuses on the agreement of data pairs. Both correlations can vary between -1 and 1 and are designed to be robust to outlier data and not assume specific distributions.

3 New Co-Symmetric Dissimilarity Functions

In [1], different similarity and dissimilarity measures that are usually used to compare distributions of probability functions are considered. They are not

Table 4. Pearson, Kendall and Spearman coefficients of comparing distances from one group.

Distance	Pearson	Kendall	Spearman
Soergel Co-Avg Vs. Soergel Co-Pro	0.9758	0.9137	0.9867
Soergel Co-Avg Vs. Sorensen Co-Avg	0.9906	0.9446	0.9933
Soergel Co-Pro Vs. Sorensen Co-Pro	0.9919	0.9613	0.9976
Soergel Co-Pro Vs. Sorensen Co-Avg	0.9656	0.8494	0.9628
Soergel Co-Avg Vs. Sorensen Co-Pro	0.9587	0.9411	0.9934
Sorensen Co Avg Vs. Sorensen Co-Pro	0.9629	0.8825	0.9766

Table 5. Pearson Kendall and Spearman coefficients of comparing distances from two group.

Distance	Pearson	Kendall	Spearman
Jaccard Co-Avg Vs. Jaccard Co-Pro	0.9566	0.8788	0.9781
Jaccard Co-Avg Vs. Dice Co-Avg	0.988	0.9375	0.9939
Jaccard Co-Pro Vs. Dice Co-Pro	0.9903	0.9634	0.9978
Jaccard Co-Pro Vs. Dice Co-Avg	0.9437	0.8164	0.9501
Jaccard Co-Avg Vs. Dice Co-Pro	0.9318	0.9153	0.9894
Dice Co-Avg Vs. Dice Co-Pro	0.9365	0.8529	0.9688

co-symmetric. We apply the method of co-symmetrization of similarity and dissimilarity functions proposed in [13] to create new dissimilarity measures of probability distributions that comply with the co-symmetry property:

$$D_{Co-Pro}(P,Q) = D(P,Q) * D\big(neg_B(P), neg_B(Q)\big), \quad (1)$$

where * is the product of real numbers. It is easy to show that the distances obtained from (1) are co-symmetric dissimilarity functions. Table 2 shows co-symmetric dissimilarity functions obtained from the four known [1] dissimilarity functions presented in Table 1.

4 Comparative Analysis of New Co-Symmetric Dissimilarity Functions

For comparative analysis of new dissimilarity measures, we used one thousand probability distributions created randomly, each with 10 elements. For the first analysis, the dissimilarity matrices were constructed for the four new co-symmetric dissimilarity measures created by equation (1).

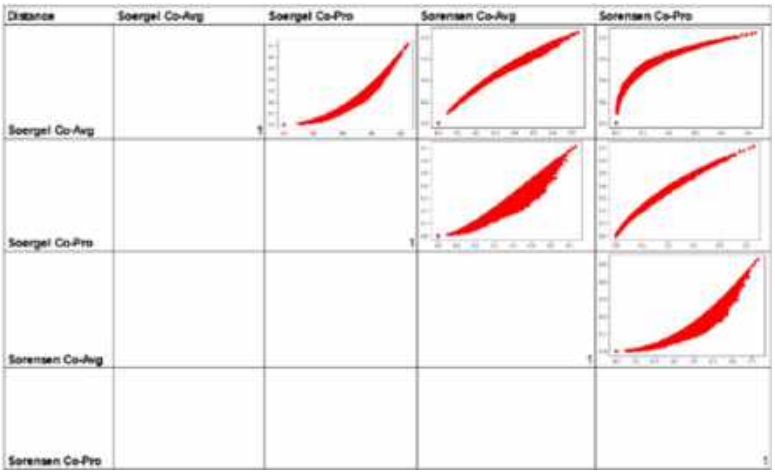
Subsequently, each dissimilarity matrix is transformed into a vector, and the correlation is calculated between two vectors corresponding to two dissimilarity matrixes obtained for two different methods.

The scatter graphs for each pair of vectors are created to graphically observe the correlation that exists between dissimilarity functions, see Fig. 1. In the same way, the correlation between the dissimilarity functions is calculated using Pearson, Kendall and Spearman correlation coefficients.

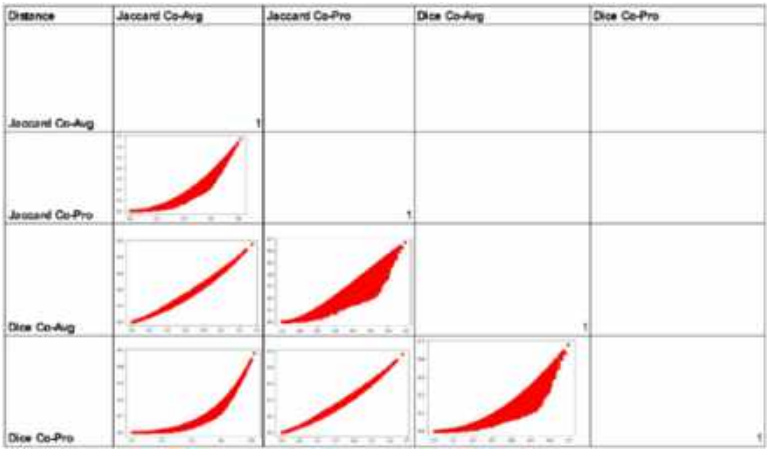
5 Comparing Similarity Functions with Higher Similarity

In [7], new co-symmetric dissimilarity measures based on average operation were obtained. It can be seen from the analysis carried out in the paper that the dissimilarity measures with the greatest correlations were two distances Sorensen Co-Avg and Soergel Co-Avg, and two distances Jaccard Co-Avg and Dice Co-Avg.

In this paper, we obtained the same result for the product-based co-symmetrization (1) of these pair of distances, see Table 3 and Fig. 1, where scatter graphs demonstrate the almost strict monotone dependence between Sorensen Co-Pro and Soergel Co-Pro distances, and between Jaccard Co-Pro and Dice Co-Pro distances.



a)



b)

Fig. 2. a) Scatter plots comparing group one and b) Scatter plots comparing group two

For further analysis of the correlation-based similarity of obtained co-symmetric distances, we divided them into two groups of most correlated distances. The first group contains co-symmetric dissimilarity functions (distances) obtained from Sorensen and Soergel distances, and the second group contains co-symmetric dissimilarity functions obtained from Jaccard and Dice distances.

The results are presented in Tables 4 and 5 and on Figures 2a) and 2b). As was expected, for most co-symmetric dissimilarity functions, different co-symmetrization of the same distance usually gives co-symmetric distances without the correlation less than 0.99. We have paid more attention to the results of the Spearman correlation, which is a measure of monotonic relationship. Only one unexpected result was obtained for Soergel Co-Avg and Sorensen Co-Pro co-symmetric dissimilarity functions, see Table 4.

6 Results

We applied the procedure of co-symmetrization based on product aggregation to the four most popular distances between probability distributions [1]. The correlation analysis of similarity between

these distances show high similarity between them with highest correlation between Soergel and Sorensen based co-symmetric distances, and between Jaccard and Dice based co-symmetric distances. These two pairs of distance are considered as two classes of similar co-symmetric distances with mutual Spearman correlation greater than 0.997 between distances from the same class.

Although we applied three correlation coefficients, Pearson, Spearman, and Kendall correlation, we paid more attention to the Spearman correlation, which is a measure of monotonic relationship. This property is important in the comparison of similarity and dissimilarity measures [14, 15].

Further, we compared co-symmetric distances in each class based on product co-symmetrization with co-symmetric distances obtained in our previous paper [7] based on average co-symmetrization of the same initial distances. The correlation between distances from the same class based on different co-symmetrization of distances is higher than 0.95.

7 Conclusion

We introduced new co-symmetric dissimilarity functions that can serve as distances between probability distributions. These dissimilarity functions take into account the symmetry of the space of finite probability distributions with respect to the uniform distribution $P_U = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)$, which is the fixed point of the negation of probability distributions defined over the set with n elements [9], such that $neg(P_U) = P_U$.

Co-symmetrization of four popular distance measures and further correlation analysis of these functions showed highest correlation between Soergel and Sorensen based co-symmetric distances, and between Jaccard and Dice based co-symmetric distances. The same results were obtained for co-symmetric distances obtained previously for another co-symmetrization method.

The obtained results give us a better understanding of known distances and co-symmetric distances obtained from them, which can be used to select suitable distances between probability distributions.

Acknowledgments

This work was partially supported by the Government of Mexico through the grant A1-S-47854 from CONACYT, Mexico, by the projects SIP 20231387 and 20240936 of Instituto Politécnico Nacional, Mexico, and by the program of developing the Scientific-Educational Mathematical Center of Volga Federal District. The authors acknowledge CONACYT for the computing resources provided through the Platform of Deep Learning for Language Technologies of the Supercomputing Laboratory of INAOE, Mexico, and acknowledge the support of Microsoft through the Microsoft Latin America Ph.D. Award.

References

1. **Cha, S. H. (2007)**. Comprehensive survey on distance/similarity measures between probability density functions. International journal of mathematical models and methods in applied sciences, Vol. 1, No. 2, pp. 1–8.
2. **Sáez, C., Robles, M., García-Gómez, J. M. (2016)**. Stability metrics for multi-source biomedical data based on simplicial projections from probability distribution distances. Statistical Methods in Medical Research, Vol. 26, No. 1, pp. 312–336. DOI: 10.1177/0962280214545122.
3. **Batyrshin, I. Z. (2021)**. Contracting and involutive negations of probability distributions. Mathematics, Vol. 9, No. 19, pp. 2389. DOI: 10.3390/math9192389.
4. **Rudas, I. J., Batyrshin, I. Z. (2023)**. Explainable correlation of categorical data and bar charts. Vol. 1, pp. 81–88. DOI: 10.1007/978-3-031-20153-0_7.
5. **Batyrshin, I. Z., Rudas, I. J., Kubysheva, N., Akhtyamova, S. (2022)**. Similarity correlation of frequency distributions of categorical data in analysis of cognitive decline severity in asthmatics. Computación y Sistemas, Vol. 26,

No. 4, pp. 1603–1609. DOI: 10.13053/cys-26-4- 4439.

6. **Batyrshin, I. Z., Tóth-Laufer, E. (2022)**. Bipolar dissimilarity and similarity correlations of numbers. Mathematics, Vol. 10, No. 5, pp. 797. DOI: 10.3390/math 10050797.
7. **Ensastegui-Ortega, M. E., Batyrshin, I., Cárdenas–Perez, M. F., Kubysheva, N., Gelbukh, A. (2024)**. Dissimilarity functions co-symmetry property: a focus on probability distributions with involutive negation. Journal of Intelligent & Fuzzy Systems, pp. 1–10. DOI: 10.3233/jifs-219363.
8. **Yager, R. R. (2015)**. On the maximum entropy negation of a probability distribution. IEEE Transactions on Fuzzy Systems, Vol. 23, No. 5, pp. 1899–1902. DOI:10.1109/tfuzz.2014. 2374211.
9. **Batyrshin, I., Villa-Vargas, L. A., Ramírez-Salinas, M. A., Salinas-Rosales, M., Kubysheva, N. (2021)**. Generating negations of probability distributions. Soft Computing, Vol. 25, No. 12, pp. 7929–7935. DOI: 10.1007/s00500-021-05802-5.
10. **Klein, I. (2022)**. Some technical remarks on negations of discrete probability distributions and their information loss. Mathematics, Vol. 10, No. 20, pp. 3893. DOI:10.3390/math 10203893.
11. **Batyrshin, I. Z. (2022)**. Fuzzy distribution sets. Computación y Sistemas, Vol. 26, No. 3, pp. 1411–1416. DOI: 10.13053/cys-26-3-4360.
12. **Batyrshin, I., Rudas, I., Kubysheva, N. (2023)**. Parametric negations of probability distributions and fuzzy distribution sets. Computación y Sistemas, Vol. 27, No. 3, pp. 619–625. DOI: 10.13053/cys-27-3- 4709.
13. **Batyrshin, I. (2019)**. Towards a general theory of similarity and association measures: similarity, dissimilarity and correlation functions. Journal of Intelligent & Fuzzy Systems, Vol. 36, No. 4, pp. 2977–3004. DOI: 10.3233/jifs-181503.
14. **Batagelj, V., Bren, M. (1995)**. Comparing resemblance measures. Journal of Classification, Vol. 12, No. 1, pp. 73–90. DOI: 10.1007/bf01202268.
15. **Omhover, J., Rifqi, M., Detyniecki, M. (2006).** Ranking invariance based on similarity measures in document retrieval. Adaptive Multimedia Retrieval: User, Context, and Feedback: AMR 2005, Revised selected papers 3, pp. 55–64. DOI: 10.1007/11670834_5.

Article received on 15/03/2024; accepted on 17/05/2024.
**Corresponding author is Ildar Batyrshin.*

ISSN 2007-9737

Use of Computer Vision Techniques for Recognition of Diseases and Pests in Tomato Plants

Ernesto García-Amaro[1,*], Jair Cervantes-Canales[1], Farid García-Lamont[1], Francisco Marcelo Lara-Viveros[2], José Sergio Ruiz-Castilla[1], Josué Espejel Cabrera[1]

[1] Universidad Autónoma del Estado de México, Centro Universitario UAEM Texcoco, Mexico

[2] Centro de Investigación en Química Aplicada, Mexico

{jcervantesc, fgarcial}@uaemex.mx, {ernestogarciaamaro, jsergioruizc}@gmail.com francisco.lara@ciqa.edu.mx, jec0309@hotmail.com

Abstract. Computer vision, for decades, has been involved in solving problems in everyday life, under the implementation of different computational methods, that have evolved over time. Feature extraction, along with other computer techniques, is considered a way to develop computer vision systems; currently, plays an important role, considered a complex task, allowing to obtain essential descriptors of the segmented images, differentiating particular characteristics between different classes, even when they share similarity with each other, guaranteeing the delivery of information not redundant to classification algorithms. Likewise, in this work, a computer vision system has been developed for the recognition of foliar damage caused by diseases and pests in tomato plants. The methodology implemented is based on four modules: preprocessing, segmentation, feature extraction, and classification; in the first module, the image is preprocessed of a color space RGB to L*a*b*; in the second module, the area interest was segmented, under the implementation of the algorithm principal component analysis PCA; in the third module, features are extracted from the area of interest, obtaining texture descriptors with the Haralick algorithm, and chromatic features through Contrast descriptors, Hu moments, Gabor characteristics, Fourier descriptors, and discrete cosine transform DCT; in the fourth module, the performance of the classification algorithms were tested, with the characteristics obtained from the previous stage, considering: SVM, Backpropagation, Logistic Regression, KNN, and Random Forests.

Keywords. Tomato diseases and pests, computer vision, feature extraction.

1 Introduction

Mexico, at present, plays a very important role in the export of different crops, sown both in protected environments (greenhouses) and in the open-air. The state of Veracruz, is the main producer of sugarcane and orange, Sinaloa of white maize, Chihuahua of yellow maize, and Sonora of wheat, grown under open-air agriculture; and the state of Chiapas, is the main producer of coffee, Guanajuato of broccoli, Mexico City of Christmas eve and Sinaloa of tomato, grown in protected environments; likewise, in Mexico, the area planted with tomatoes is 42 383.3 hectares, obtaining a yield of 2 860 305.19 tons of production annual [23]. Currently, the methods in agriculture have evolved, achieving an increase in the production per plant and quality of the fruit; these results have been obtained with the implementation of new automated techniques in

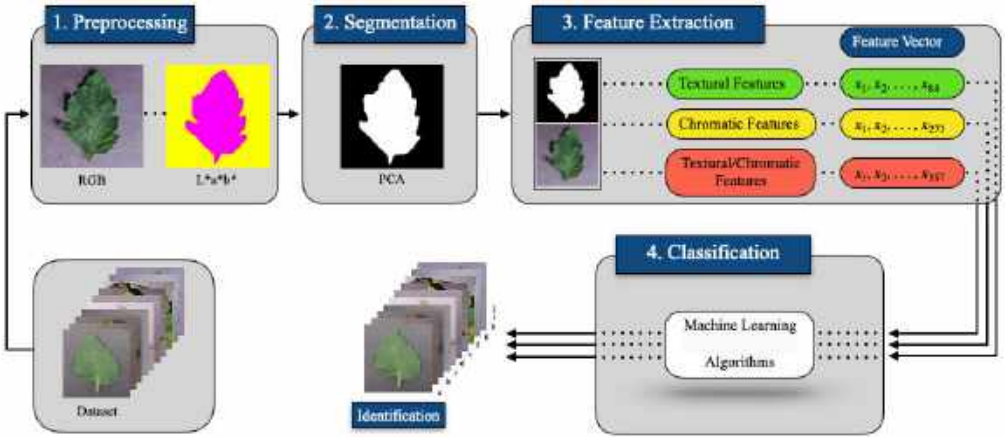


Fig. 1. Proposed method

the crops, both in the open-air and in protected environments, developing tasks in the care of planting, nutrition, growth, and harvesting of the same.

Over time, has exponentially increased the production of different crops, obtaining considerable financial income in some entities of the country, however, there are some risks in the cultivation process. Producers have reported economic declines, due to diseases y pests that have attacked tomato plants (Lycopersicum esculentum), or even, totally contaminated crops, reflecting financial losses.

Some of the most common diseases in tomato plants, the following are considered: root rot, bacterial cancer of the tomato, freckle and bacterial spot, leaf mold, gray mold, early blight, late blight, and dusty ashes [4], presented by variations of humidity, drought, temperature, residues of previous crops, wind, insects, overcast and negligence of crop operators; likewise, some of the pests more common, such as: whiteflies, leafminers, tomato psyllid, spider mites two-spotted and thrips [27], presented by variations of temperature, dust, sandy ground, humidity, inter alia; both diagnosed, through the root, stem, leaf or fruit. After the identification of an anomaly in the plant, the producer turns to experts to diagnose the disease or pest, which is considered a late detection and inaccurate; likewise, the recommended dose of a pesticide or fungicide is applied to control and/or eliminate it, generating additional expenses; in the worst-case scenario, the plants are identified with the risk that neighboring crops will be infected; therefore, contaminated plants are completely removed to prevent spread.

The main cause of loss of tomato production, is the wrong recognition of pests and diseases, since in some cases experts in the agricultural area, perform the ocular shape detection, considered an inaccurate method; for this reason, computer vision algorithms have recognized precisely foliar damage, caused by: leaf mold, late blight, early blight, bacterial spot, septoria leaf spot, target spot, tomato mosaic virus, tomato yellow leaf curl virus, spider mites two-spotted and a completely healthy class, in tomato plant leaves, avoiding the excessive or wrong application of chemical products, reducing the impact on plants and humans, in addition, contributing in the decreased loss by production, reducing financial hurt.

2 Related Work

Computer sciences, recently, have been involved in solving problems in various multidisciplinary issues, in which, the existence of living beings on planet earth becomes more stable, allowing to alert, identify or predict catastrophes that affect the environment in the one we live; in the existing literature, there are investigations with very

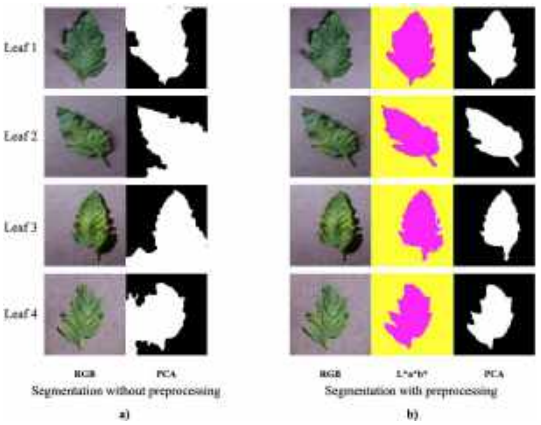


Fig. 2. Preprocessing and segmentation

Fig. 3. Dataset plantvillage

promising results, however, the ceiling has not yet been reached and there is a great opportunity to contribute to the scientific field. The plants, in their variety of genus, are currently of great importance, since they have a fundamental role for all living beings in their entire environment.

In this part of the manuscript, the works related to this research are described, all of them focused on the agricultural area, solving issues, such as: classification and recognition of leaves, and identification of diseases and pests in plants through the leaf, implementing techniques of digital image processing, image segmentation, feature extraction, machine learning algorithms, deep learning, etc.

In the literature, exhaustive studies of works have been carried out with various methodologies, applicable to detect and classify diseases in leaves of different plants, using computer vision techniques [12]; likewise, researchers have contributed to the field of color image segmentation [18], considered a field that to this day is rigorously studied, both in controlled and uncontrolled environments, being a subject with great impact, since it influences on feature extractors and in the performance of the classification algorithms; on the other hand, under the implementation of modified fully-convolutional networks FCNs, it has been possible to segment images of plants through the leaf [36].

In previous investigations, works have been developed for the identification and classification of plants through the leaf, in [25, 40] have developed proposals methodological with deep learning techniques, specifically, convolutional neural networks CNN, comparing the performance with the architectures existing; likewise, in [7, 8, 24, 3] techniques of extraction and selection of characteristics have been implemented, considering color, shape, and texture, classifying with machine learning algorithms, obtaining favorable results for the same purpose.

In the country and in many parts of the world, the crops are affected by the unwanted arrival of pests [19] and diseases [38], both in protected environments and outdoors, likewise, this has a direct impact on production, reducing the producers financial balances; therefore, in [29] they have developed a system for the detection of diseases in different plants, using characteristics extraction techniques with Gabor wavelet transform GWT and SVM for classification; on the other hand, in [30, 28] digital image processing and machine learning methods were implemented for the recognition of diseases in tomato plant leaves.

With scientific advances and the development of new computational methods to solve problems in the field of object recognition in images, deep learning, in essence, convolutional neural networks CNN has positioned itself among the most used today, likewise, networks CNN have been evaluated for the detection of diseases and pests in tomato plants [15]; furthermore, deep learning and machine learning techniques have been merged for the same purpose [33]. In the literature, deep learning has had a great

Table 1. Dataset information

Class	Disease or pest common name	Disease or pest scientific name	Images number
a	Tomato mosaic virus	Tomato mosaic virus (ToMV)	373
b	Leaf mold	Fulvia fulva	952
c	Early blight	Alternaria solani	1000
d	Target spot	Corynespora cassiicola	1404
e	Healthy	Completely healthy leaves	1591
f	Spider mites two-spotted	Tetranychus urticae	1676
g	Septoria leaf spot	Septoria lycopersici	1771
h	Late blight	Phytophthora infestans	1908
i	Bacterial spot	Xanthomonas campestris pv. vesicatoria	2127
j	Tomato yellow leaf curl virus	Begomovirus (Fam. Geminiviridae)	5357

boost, since research has been carried out under this scheme.

With the implementation of the CNNs, has been evaluating and monitoring each proposed architecture, for the detection and recognition of diseases in tomato plant through of the leaves [2, 14, 17, 34, 37, 38, 39]; finally, and without leaving behind, in [32] a robotic system has been developed in conjunction with artificial vision techniques in greenhouses for the same purpose.

3 Materials and Methods

This section, presents the methodological proposal for this research, in which a system with four stages is exposed, preprocessing, segmentation, feature extraction, and classification; likewise, the dataset used is described. The adopted method, develops tasks such as: transformation from one color space to another, obtaining the area of interest, and the extraction of textural and chromatic features, in addition, through machine learning algorithms, has been achieved to identify foliar damage caused by diseases and pests in tomato plant leaves; contributing to the reduction of financial losses and the excessive or wrong application of chemical products in crops, decreasing their consumption in humans and plants. In Fig. 1, the implemented methodology is displayed.

3.1 Preprocessing

In stage 1 in Fig. 1 of the proposed methodology, the images of the dataset used are preprocessed, which consists of a transformation from RGB color space to L*a*b* color space.

The intensity of the different color components in RGB, determine both the tone and the brightness, in addition, it is an optimal format for the visualization of color in electronic equipment such as television and image acquisition equipment, however, it is not best suited for color image processing or segmentation, due to the high correlation between R, G and B components.

Therefore, for this research, the L*a*b* [22] color space has been used, defined by three variables: L* is the intensity, a* and b* the tonality components, the placement of this color space is similar to RGB space, but the position of the variables is different.

3.2 Segmentation

After the preprocessing of stage 1, the images have been segmented, executing the algorithm principal component analysis PCA [32], obtaining, as a result, the area of interest, which is will analyze in the next stage, determining the edges and calculating its properties, extracting textural and chromatic characteristics, and the combination of both, textural/chromatic.

Table 2. Performance of algorithms for classification

Classifier	Textural	Chromatic	Textural Chromatic
KNN	74.95	82.67	84.13
Logistic Regression	73.95	83.95	86.05
Random Forests	77.91	85.12	86.63
Backpropagation	81.83	90.65	83.76
SVM	**89.40**	**93.69**	**94.46**

Due to the nature of the dataset used, which was is in an RGB color space, the segmentation stage was supported by a previous preprocessing, which helped to competently segment the images. In Fig. 2, four tests are displayed, for four different sheets; in part a), tests were made by directly segmenting in RGB color space with the PCA algorithm, doing it incorrectly; in part b), the same tests were carried out, but before the segmentation a preprocessing was applied to the images, transforming from the RGB color space to the L*a*b* color space; likewise, it is concluded that the implemented segmentation method has a better performance by applying a previous preprocessing stage.

3.3 Feature Extraction

In this section of the manuscript, the process and techniques used to extract the characteristics of each of the images in the dataset are described, considered a delicate process and a fundamental pillar for the next stage of the proposed method; the characteristics obtained in this work are invariant to scaling, rotation, and translation, which allows the classifier to recognize objects despite their size, orientation, and position.

Likewise, an analysis has been carried out with two characteristics extraction techniques, considering, textural features, chromatic features, and the combination of both, textural/chromatic features, getting descriptors with high discriminative power, representing each image through numerical values, later, in the next stage of the proposed system, the characteristic vectors obtained are evaluated with machine learning algorithms.

3.3.1 Textural Features

The texture characteristics of a leaf, are obtained from the surface, through the area of interest generated in the second stage of the proposed methodology. The textural feature extraction algorithms, look for basic repeating patterns with periodic or random structures in images.

The texture is manifested in properties such as: roughness, harshness, granulation, fineness, smoothness, among others; likewise, it is invariant to displacements, since it repeats a pattern across a surface, therefore, it is explained because the visual perception of a texture is independent of position.

In this work, Haralick feature extractors have been implemented [20], taking into account the distribution of intensity values in the region, obtaining the mean and range of the following variables: mean, median, variance, smoothness, bias, kurtosis, correlation, entropy, contrast, homogeneity, etc; calculated as follows:

$$f_1 = \sum_i \sum_j \left[p(i,j)^2\right], \tag{1}$$

$$f_2 = \sum_{n=0}^{N_g-1} n^2 \left\{ \begin{array}{c} \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j) \\ |i-j|=n \end{array} \right\}, \tag{2}$$

$$f_3 = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_g}\left[ijp(i,j) - \mu_x\mu_y\right]}{\sigma_x\sigma_y}, \tag{3}$$

$$f_4 = \sum_i \sum_j (i-\mu_x)^2 p(i,j), \tag{4}$$

$$f_5 = \sum_i \sum_j \frac{1}{1+(i-j)^2} p(i,j), \tag{5}$$

$$f_6 = \sum_{i=2}^{2N_g} iP_{x+y}(i), \tag{6}$$

$$f_7 = \sum_{i=2}^{2N_g} (i-f_8)^2 P_{x+y}(i), \tag{7}$$

$$f_8 = -\sum_{i=2}^{2N_g} P_{x+y}(i)\log\{P_{x+y}(i)\}, \tag{8}$$

$$f_9 = -\sum_i \sum_j p(i,j)\log\{p(i,j)\}, \tag{9}$$

Table 3. Performance metrics evaluated

Metric	Formula
Accuracy	$\text{Acc} = \frac{(TP+TN)}{(TP+TN+FP+FN)}$
Precision	$\text{Precision} = \frac{TP}{(TP+FP)}$
Recall	$\text{Recall} = \frac{TP}{(TP+FN)}$
F-Measure	$\text{F-Measure} = \frac{(2*precision*recall)}{(precision+recall)}$
FP Rate	$\text{FP Rate} = \frac{FP}{(FP+TN)}$
MCC	$\text{MCC} = \frac{(TP*TN)-(FP*TN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

$$f_{10} = \sum_{i=0}^{N_g-1} (i-f_8)^2 P_{x-y}(i), \tag{10}$$

$$f_{11} = -\sum_{i=0}^{N_g-1} P_{x-y}(i)\log\{P_{x-y}(i)\}, \tag{11}$$

$$f_{12} = \frac{HXY - HXY1}{\max\{HX, HY\}}, \tag{12}$$

$$f_{13} = (1-e^{[-2(HXY2-HXY)]})^{\frac{1}{2}}, \tag{13}$$

$$f_{14} = (\text{Second largest eigenvalue } of\ Q)^{\frac{1}{2}}. \tag{14}$$

The vector of textural characteristics obtained X_t, can be represented as: $X_t = [x_1, x_2, \ldots, x_{84}]$. Where, the numerical value of $[x_1, x_2, \ldots, x_{28}]$ are from the R component, of $[x_{29}, x_{30}, \ldots, x_{56}]$ belongs to the G component, and of $[x_{57}, x_{58}, \ldots, x_{84}]$ are from the B component. The values of the R, G, and B components are concatenated, forming the vector X_t.

3.3.2 Chromatic Features

The chromatic characteristics, provide relevant information of a portion of the image that has been segmented, the exhaustive analysis carried out by this type of techniques, is done starting from a specific color space, for example: extracting information from the primary color channels, like: red, green and blue RGB; hue, saturation, and value HSV, L*a*b*, etc. The algorithms, Contrast descriptors [13], gabor characteristics [16, 29], Hu moments [21], discrete cosine transform DCT [9, 10], and Fourier descriptors [26], were implemented for the extraction of chromatic characteristics, calculating the descriptors of all the images in the dataset. The Contrast descriptors of an image, define information about the difference in intensity between a region and its neighborhood. The smaller the difference, the lower the contrast. Contrast is defined as follows:

$$K_1 = \frac{G - G_e}{G_e}, \tag{15}$$

$$K_2 = \frac{G - G_e}{G + G_e}, \tag{16}$$

$$K_3 = ln(G/G_e), \tag{17}$$

where G and G_e denote the mean value in the region and in the neighborhood respectively.

The Gabor characteristics, it is considered another robust technique, used for the extraction of features in images; being a hybrid technique, composed of the nucleus of the Fourier transformation on a Gaussian function; also, the frequency resolution is more sophisticated than other techniques, since the Gaussian signal is more concentrated than the rectangular function in the frequency domain. Gabor transformation is a 2D filter, represented by the following equation:

$$G(t,w) = \int_{-\infty}^{\infty} e^{-\frac{(\tau-t)^2}{2}} e^{-jwt} x(\tau) d\tau. \tag{18}$$

On the other hand, the implementation of the seven Hu moments, they have managed to integrate information of the variable of the color of the area of interest; calculated as follows:

$$\phi_1 = \eta_{20} + \eta_{02}, \tag{19}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \tag{20}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2, \tag{21}$$

$$\phi_4 = (\eta_{30} - 3\eta_{12})^2 + (\eta_{21} + \eta_{03})^2, \tag{22}$$

$$\begin{aligned}\phi_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 \\ & -3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \\ & [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2],\end{aligned} \tag{23}$$

$$\begin{aligned}\phi_6 = {} & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & +4(\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})),\end{aligned} \tag{24}$$

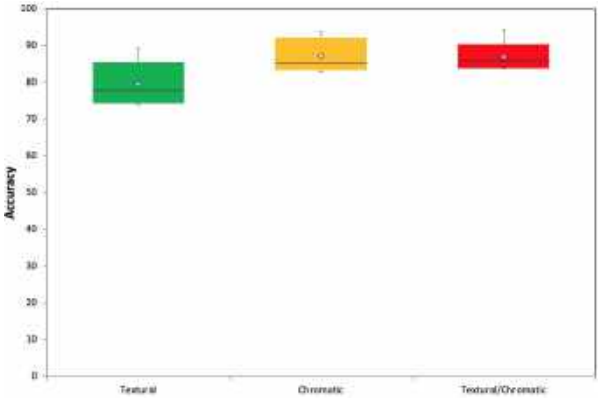


Fig. 4. Accuracy of feature extraction techniques

$$\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]. \quad (25)$$

Likewise, the discrete cosine transform DCT, contributes to the generation of extraction of features chromatic; the DCT uses base transformations and cosine functions of different wavelengths.

A particularity about DCT in relation to the discrete Fourier transform DFT, is the limitation to the use of real coefficients. The DCT in two dimensions, is derived directly from the definition of the one-dimensional case, thus, it is calculated as follows:

$$F(u,v) = \frac{2}{\sqrt{MN}} \sum_{x=1}^{N} \sum_{y=1}^{M} I(x,y) \cdot c_u \cdot \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cdot c_v \cdot \cos\left(\frac{\pi(2y+1)v}{2M}\right) = \frac{2c_u c_v}{\sqrt{MN}} \sum_{x=1}^{N} \sum_{y=1}^{M} I(x,y) \cdot E_u^N(x) \cdot E_v^M(y). \quad (26)$$

Finally, other characteristics were obtained with the Fourier descriptors, calculated using the following equation: $d_u = |F(u)|$; where $F(u)$ is calculated for $u = 1, \ldots, N$, where N is the number of descriptors to calculate; also, for a two-dimensional function, in the case of an image $I(x,y)$ of size MxN, they are defined as:

$$F(u,v) = \frac{1}{\sqrt{MN}} \sum_{y=0}^{M-1} \sum_{x=0}^{N-1} I(x,y) \cdot e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}. \quad (27)$$

After the execution of the various algorithms for extraction of chromatic characteristics, the resulting numerical vector for each image, has a length of 273, represented by the next equation: $X_t = [x_1, x_2, \ldots, x_{273}]$; where, the contrast descriptors provide 15 characteristics, considering $[x_1, x_2, \ldots, x_{15}]$; the Gabor characteristics provides 201 features, considering $[x_{16}, x_{17}, \ldots, x_{216}]$; the Hu moments add 21 characteristics, considering $[x_{217}, x_{218}, \ldots, x_{237}]$; the DCT provides 12 characteristics, considering $[x_{238}, x_{239}, \ldots, x_{249}]$; and finally, the Fourier descriptors provide 24 characteristics, considering $[x_{250}, x_{251}, \ldots, x_{273}]$, concatenated in the vector X_t.

Likewise, tests were developed combining the textural and chromatic characteristics, obtaining as a result, a numerical value of 357 characteristics for each image of the dataset. The vector of texture features $[x_1, x_2, \ldots, x_{84}]$, it has been concatenated with the vector of chromatic features $[x_1, x_2, \ldots, x_{273}]$, obtaining as a result a hybrid vector, represented by: $X_t = [x_1, x_2, \ldots, x_{357}]$.

3.4 Classification

Finally, in stage number four of the methodology, machine learning algorithms have been used to recognize ten different classes, likewise, measuring performance with, Support Vector Machines SVM, Backpropagation, K-Nearest Neighbors KNN, Random Forests, and Logistic Regression, tested with different feature extraction techniques.

In the experiments carried out, cross-validation with $k = 10$ was used to validate results, that is, 10 tests were performed with 90 % and 10 % of the data for training and testing respectively. A brief, description of the machine learning algorithms used is given below.

3.4.1 K-Nearest Neighbors KNN

The KNN algorithm, classifies a new point in the dataset, based on euclidean distance, finding the k closest distances to the object to classify. As the first instance, distances from the new point to each object in the dataset are calculated, the euclidean

Table 4. Performance by class of the tested algorithms

Class	KNN	Logistic Regression	Random Forests	Back-propagation	SVM
a	0.863	0.798	0.942	0.903	0.920
b	0.794	0.808	0.875	0.884	0.903
c	0.674	0.652	0.721	0.721	0.805
d	0.700	0.772	0.787	0.848	0.907
e	0.949	0.955	0.946	0.963	0.988
f	0.949	0.821	0.803	0.871	0.932
g	0.792	0.794	0.836	0.869	0.935
h	0.813	0.759	0.786	0.838	0.905
i	0.839	0.895	0.887	0.944	0.967
j	0.928	0.952	0.927	0.976	0.989

distance of a point a to a point b it is calculated as follows:

$$d(P_0, P_1) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \quad (28)$$

Subsequently, are located the k closest distances to the new point, finally, the class of the closest point in the dataset is assigned by majority vote. For further analysis of the KNN algorithm, refer to [1].

3.4.2 Logistic Regression

Logistic regression, is used to model the posterior class probabilities, without having to learn the conditional class densities, facilitating the classification into small training sets and less complexity.

$\pi_i = p(Y_i = 1|X_i)$ where X_i is a vector of size $1 * (p + 1)$ with the first element equal to 1, and the remaining elements, corresponding to the characteristics extracted from the leaf for the example i. The logistic regression model relates π_i with the characteristics using the function:

$$logit(\pi_i) = log(\frac{\pi_i}{1 - \pi_i}) = X_i\beta, \quad (29)$$

where $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ is the vector of regression coefficients. For a more in-depth study, refer to [5, 11].

3.4.3 Random Forests

Random Forests, is an algorithm composed of decision tree classifiers, each tree depends on the values of a random vector con with sampling independently and with the same distribution for all trees in the forest. Generalization error for forests converges to a limit, as the number of trees in the forest increases.

When a model is generalized and fails, depends on the strength of individual trees in the forest and the correlation between them. By randomly selecting features to divide each node, error rates occur that compare favorably with the Adaboost algorithm, but are more robust with respect to noise.

In [6], the Random Forests algorithm is described, specifying the characterization of precision, the use of random characteristics, the selecting random entries, the linear combination inputs, the Adaboost algorithm operation, the effects of output noise, the weak data inputs, the random forests for regression, theorems, and equations that lead to the execution of the Random Forests classifier.

3.4.4 Backpropagation

Artificial neural networks ANN, nowadays, try to imitate the learning process and solution of the human brain, this is achieved with the implementation of computational methods applied to different areas.

Humans, to solve problems of daily life, take prior knowledge, acquired from the experience of some specific area, likewise, artificial neural networks, collect information on solved problems to build models or systems that can make decisions automatically.

The multiple connections between neurons, form an adaptive system, the weights of which are updated using a particular learning algorithm. One of the most used algorithms and the one that was implemented in this work, was the algorithm of backpropagation BP; which in general, performs the learning and classification process in four points, initialization of weights, forward spread, backward spread, and the updating of weights.

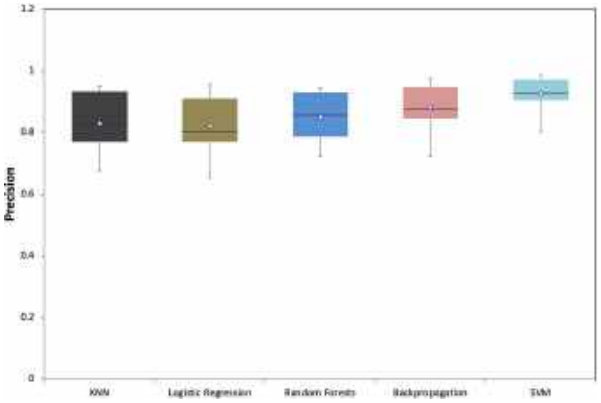


Fig. 5. Graphics of boxplots by algorithm

To carry out the learning process, backpropagation algorithm iteratively changes weights between neurons, minimizing the quadratic error between the desired output and that obtained with the current weights.

Each of the training set examples $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ are used to adjust the weights in the network. By being presented an example, the signal is propagated forward in the network until the output is obtained. The output of the j-th hidden unit is calculated as:

$$o^h_{nj} = f^h_j(net^h_{nj}) = \frac{1}{1 + exp(-net^h_{nj})}, \qquad (30)$$

where $net^h_{nj} = \sum w^h_{ji} x_{ni} + \theta^h_j . w^h_{ji}$ is the weight of the connection of the i-th input neuron to the j-th hidden neuron. θ^h_j and f^h_j represent the bias and the activation function of the j-th hidden neuron. So, the output of the k-th neuron is represented by:

$$o^o_{nk} = f^o_k(net^o_{nj}) = \frac{1}{1 + exp(-net^o_{nk})}, \qquad (31)$$

where the superscripts h and o they refer to the quantities in the hidden and output layers respectively. For a more in-depth study of the algorithm, refer to [31].

3.4.5 Support Vector Machines SVM

The main characteristics that identify the SVM algorithm, are the use of kernels when working in non-linear sets, the absence of local minima, depends on a small subset of data and the discriminative power of the model constructed by optimizing the separability margin between the classes.

SVM is a linear classifier, in other words, it classifies between two data sets through the construction of a line that separates two classes. When this is not possible, a function called *Kernels* is used, which transforms the input space to a highly dimensional space, where the sets can be linearly separated after the transformation.

However, the choice of a function is restricted to those that satisfy the Mercer conditions. Training an SVM allows solving a quadratic programming problem, as shown below:

$$\max_{\alpha_i} -\frac{1}{2}\Sigma^l_{i,j=1}\alpha_i y_i \alpha_j y_j \mathbf{K}\langle x_i \cdot x_j\rangle + \Sigma^l_{i=1}\alpha_i. \qquad (32)$$

subject to: $\Sigma^l_{i=1}\alpha_i y_i = 0$, $C \geq \alpha_i \geq 0, i = 1, 2, \ldots, l$,
where $C > 0, \alpha_i = [\alpha_1, \alpha_2, \ldots, \alpha_l]^T, \alpha_i \geq 0, i = 1, 2, \ldots, l$ are coefficients that correspond to x_i, y_i with α_i nonzero which are called Support Vectors SV. For a more in-depth study of the algorithm, refer to [35].

3.5 Dataset

The images used in this investigation, belong to the Plantvillage dataset [14, 33, 34, 38], which has been acquired through an Internet repository of free environment; considering ten different classes, eight diseases (class a, b, c, d, g, h, i, and j), one pest (class f) and one completely healthy class (class e), the images are in an RGB color space, with dimensions of 256x256 pixels, see Table 1, and visually relate it to Fig. 3.

4 Results and Discussions

In this section of the manuscript, the metrics used are defined and the experimental results of the tests developed are analyzed and discussed. The results obtained are visualized with tables, confusion matrices, and boxplots, through exit percentages of the performance of classifiers, for each algorithm used, accuracy and precisions by class are reported.

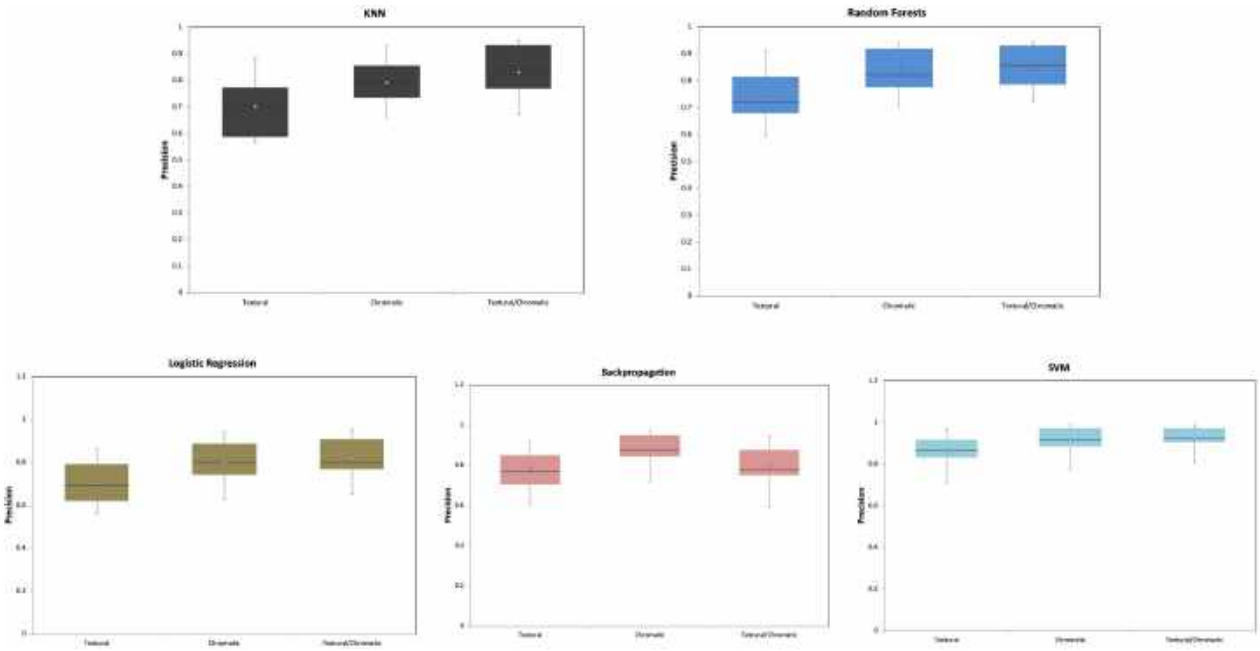


Fig. 6. Performance of used algorithms

4.1 Performance Metrics

Accuracy, Precision, Recall, F-Measure, FP Rate, MCC, are the metrics evaluated for the experimental results presented in this work, defined in Table 3.

4.2 Experimental Results

Table 2, shows the results of the algorithms used, evaluating performance against the feature extraction techniques mentioned in the proposed method.

The algorithm that obtained the lowest percentage of correctly classified instances, was KNN, with 82.67% for the chromatic characteristics, and for the combination of both, textural/chromatic an 84.13% respectively, nevertheless, for the test with textural features, has outperformed the algorithm Logistic Regression with 74.95%.

The Logistic Regression algorithm, was the second with lower results, outperforming to classifier KNN in tests with chromatic features and the combination of both, textural/chromatic; obtaining an 83.95% of accuracy for chromatic characteristics, and for the combination of both, textural/chromatic an 86.05%. KNN classifier and Logistic Regression had very similar behavior in their performance, however, the third-best algorithm, was Random Forests, obtaining 77.91% for textural features, for chromatic features, and textural/chromatic hybrid characteristics, it is has exceeded 85% respectively.

One of the classifiers with the best performance for this research, was the Backpropagation learning algorithm for artificial neural networks; with an accuracy percentage of 81.83% for textural characteristics, for chromatic characteristics, a 90.65% was obtained, and finally for hybrid, textural/chromatic characteristics it had the lowest performance than its counterparts, with an 83.76%.

The best performance for the proposed system, was obtained by the SVM algorithm, achieving an accuracy of 89.40% for textural features, for chromatic characteristics, 93.69% was obtained, finally, for hybrid, textural/chromatic features, was demonstrated a 94.46% respectively.

Table 5. Confusion matrix for backpropagation

	a	b	c	d	e	f	g	h	i	j
a	344	4	1	2	0	10	9	1	0	2
b	8	818	22	5	3	6	32	37	4	17
c	5	16	682	44	3	27	49	113	30	31
d	5	7	33	1158	24	103	27	20	16	11
e	1	2	0	24	1544	10	4	5	1	0
f	11	4	23	73	7	1506	17	21	1	13
g	6	23	20	23	8	13	1586	61	20	11
h	1	40	113	24	12	19	62	1602	20	15
i	0	2	23	4	3	2	22	26	2015	30
j	0	9	29	8	0	33	18	25	28	5207

In Table 2, for each classifier tested, the best results have been achieved based on the extraction of the hybrid, textural/chromatic features, except with the backpropagation learning algorithm, since the best performance has been obtained with characteristics chromatic.

In the boxplots of Fig. 4, it is notable, that the performance of the hybrid, textural/chromatic characteristics (red boxplot) considerably surpasses the textural features (green boxplot), and slightly the chromatic features (yellow boxplot).

In Fig. 4, 5, and 6, the results of each of the tests carried out with the classification algorithms and feature extraction methods are displayed, plotted using box plots, where the data distribution is analyzed, considering the median, value minimum, maximum, and intermediate. In Table 4, the best performance is reflected for each of the tested algorithms, showing the precisions by class, likewise, are graphed in Fig. 5.

The best two precisions of the tested algorithms exceed 0.94%; obtaining 0.942% for class (a) and 0.946% for class (e) with Random Forests; for class (e) and (f) 0.949% was achieved with KNN; for class (j) a 0.952% and 0.955% were obtained for class (e) with Logistic Regression; with the Backpropagation algorithm, 0.963% was achieved for class (e) and 0.976% for class (j); finally, for class (e) a 0.988% was obtained and for class (j) a 0.989% with SVM, see Table 4.

In Fig. 5, the boxplot of the algorithm KNN and Logistic Regression, have the largest data ranges, so the values are more dispersed or separated from their counterparts; likewise, the precision data ranges of the Random Forests and Backpropagation algorithm are moderately more concentrated, that the KNN and Logistic Regression algorithm; finally, the algorithm with the best performance was the SVM classifier, since the data is more concentrated compared to the other tests.

In Table 4, are plotted the precisions by class, obtained from the experimentation with the machine learning algorithms front to the features extraction methods. For each algorithm used, tests were carried out with textural features, chromatic features, and textural/chromatic hybrid features.

From the experimentation developed, the algorithm that showed the lowest performance was KNN, in addition, the precisions obtained are more dispersed in comparison with the rest of the classifiers. However, the algorithm with the best results was SVM, since the data is more concentrated than those of its counterpart.

In Table 5 and 6, confusion matrices are shown two, considering the performance of the two best classifiers, highlighting the backpropagation algorithm and SVM, likewise, an analysis of confusion between the ten classes is made. For an understanding of the confusion matrices, the nomenclature is as follows, where: a=Tomato mosaic virus, b=Leaf mold, c=Early blight, d=Target spot, e=Healthy, f=Spider mites, g=Septoria leaf spot, h=Late blight, i=Bacterial spot, and j=Tomato yellow leaf curl virus.

The matrix of the Table 5, has been built from the tests performed with the Backpropagation algorithm front to chromatic features, the analysis by class is the following: for class a, the model has confused more with the class (f); for the class b, the confusion highest was with the class (h); for the class c, the confusion highest was with the class (h); for the class d, the confusion highest was with the class (f); for the class e, the confusion highest was with the class (d); for the class f, the confusion highest was with the class (d); for the class g, the confusion highest was with the class (h); for the class h, the confusion highest was with the class (c); for the class i, the confusion highest was with the class (j); and for the class j, the confusion highest was with the class (f). The matrix of the Table 6, has been built from the tests performed with the algorithm SVM, front to chromatic features, the analysis by class is the following: for class a,

Table 6. Confusion matrix for SVM

	a	b	c	d	e	f	g	h	i	j
a	352	6	0	1	0	6	5	2	0	1
b	5	877	14	2	0	3	19	25	2	5
c	6	19	783	34	1	9	17	103	13	15
d	2	1	24	1270	3	69	11	10	7	7
e	0	1	3	15	1563	1	3	5	0	0
f	8	4	14	75	1	1552	4	10	0	8
g	13	24	20	20	0	4	1639	32	9	10
h	4	35	114	9	7	13	35	1665	14	12
i	0	2	20	6	2	0	5	17	2054	21
j	0	10	21	8	0	14	7	12	26	5259

the model has confused more with the class (b,f); for the class b, the confusion highest was with the class (h); for the class c, the confusion highest was with the class (h); for the class d, the confusion highest was with the class (f); for the class e, the confusion highest was with the class (d); for the class f, the confusion highest was with the class (d); for the class g, the confusion highest was with the class (h); for the class h, the confusion highest was with the class (c); for the class i, the confusion highest was with the class (j); and for the class j, the confusion highest was with the class (i). In most of the experimental tests, the class that showed the most confusion and the one that the models assigned as correct, was class h, belonging to late blight disease.

5 Conclusions

In this manuscript, work was developed based on features extraction techniques and machine learning, for the recognition of foliar damage caused by pests and diseases that affect tomato plants. After preprocessing and image segmentation, the proposed system extracts textural features, chromatic features, and the features hybrid, textural/chromatic, finally, automatic learning algorithms evaluate the obtained descriptors.

Derived from the tests in the preprocessing and segmentation stage, it is verified that the implemented segmentation method, has a better performance by applying a previous preprocessing stage; likewise, of the three characteristics extraction methods implemented in this research, the one that obtained the best descriptors, directly impacting on the performance of the classifiers, were the features hybrid, textural/chromatic; furthermore, the best classifier was SVM; therefore, it was shown, that by applying the image color space transformation of input, the segmentation PCA method, the conjunction of textural/chromatic feature extraction, and the SVM classification process, the system has achieved a performance favorably.

References

1. **Aha, D. W., Kibler, D., Albert, M. K. (1991).** Instance-based learning algorithms. Machine Learning, Vol. 6, pp. 37–66. DOI: 10.1007/bf00153759.
2. **Ahmad, I., Hamid, M., Yousaf, S., Shah, S. T., Ahmad, M. O. (2020).** Optimizing pretrained convolutional neural networks for tomato leaf disease detection. Complexity, Vol. 2020, pp. 1–6. DOI: 10.1155/2020/8812019.
3. **Ayala-Niño, D., Ruíz-Castilla, J. S., Arévalo-Zenteno, M. D., Jalili, L. D. (2019).** Complex leaves classification with features extractor. Intelligent Computing Theories and Application, Springer International Publishing, pp. 758–769. DOI: 10.1007/978-3-030-26969-2_72.
4. **Blancard, D. (2011).** Enfermedades del tomate. Mundi-Prensa.
5. **Borges, J. S., Bioucas-Dias, J. M., Marcal, A. R. (2011).** Bayesian hyperspectral image segmentation with discriminative class learning. IEEE Transactions on Geoscience and Remote Sensing, Vol. 49, No. 6, pp. 2151–2164. DOI: 10.1109/tgrs.2010.2097268.
6. **Breiman, L. (2001).** Random forests. Machine learning, Vol. 45, pp. 5–32. DOI: 10.1023/a:1010933404324.
7. **Cervantes, J., Garcia-Lamont, F., Rodriguez-Mazahua, L., Zarco-Hidalgo, A., Ruiz-Castilla, J. S. (2018).** Complex identification of plants from leaves.

Intelligent Computing Methodologies: 14th International Conference, ICIC 2018, Springer International Publishing, pp. 376–387. DOI: 10.1007/978-3-319-95957-3_41.

8. **Cervantes, J., Taltempa, J., García-Lamont, F., Ruiz-Castilla, J. S., Yee-Rendon, A., Jalili, L. D. (2017).** Análisis comparativo de las técnicas utilizadas en un sistema de reconocimiento de hojas de planta. Revista Iberoamericana de Automática e Informática Industrial RIAI, Vol. 14, No. 1, pp. 104–114. DOI: 10.1016/j.riai.2016.09.005.
9. **Cuevas, E., Zaldívar, D., Pérez, M. (2016).** Procesamiento digital de imágenes con MATLAB & Simulink. Ra-Ma.
10. **Dabbaghchian, S., Ghaemmaghami, M. P., Aghagolzadeh, A. (2010).** Feature extraction using discrete cosine transform and discrimination power analysis with a face recognition technology. Pattern Recognition, Vol. 43, No. 4, pp. 1431–1440. DOI: 10.1016/j.patcog.2009.11.001.
11. **Dempster, A. P., Laird, N. M., Rubin, D. B. (1977).** Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B (Methodological), Vol. 39, No. 1, pp. 1–22. DOI: 10.1111/j.2517-6161.1977.tb01600.x.
12. **Dhingra, G., Kumar, V., Joshi, H. D. (2017).** Study of digital image processing techniques for leaf disease detection and classification. Multimedia Tools and Applications, Vol. 77, No. 15, pp. 19951–20000. DOI: 10.1007/s11042-017-5445-8.
13. **Donis-González, I. R., Guyer, D. E., Pease, A. (2016).** Postharvest noninvasive classification of tough-fibrous asparagus using computed tomography images. Postharvest Biology and Technology, Vol. 121, pp. 27–35. DOI: 10.1016/j.postharvbio.2016.07.012.
14. **Durmus, H., Gunes, E. O., Kirci, M. (2017).** Disease detection on the leaves of the tomato plants by using deep learning. 2017 6th International Conference on Agro-Geoinformatics, IEEE, pp. 1–5. DOI: 10.1109/agro-geoinformatics.2017.8047016.
15. **Fuentes, A., Yoon, S., Kim, S. C., Park, D. S. (2017).** A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. Sensors, Vol. 17, No. 9, pp. 2022. DOI: 10.3390/s17092022.
16. **Gabor, D. (1946).** Theory of communication: Journal of the institute of electrical engineers. Vol. 93, pp. 429–457.
17. **Gadekallu, T. R., Rajput, D. S., Reddy, M. P. K., Lakshmanna, K., Bhattacharya, S., Singh, S., Jolfaei, A., Alazab, M. (2020).** A novel PCA–whale optimization-based deep neural network model for classification of tomato plant diseases using GPU. Journal of Real-Time Image Processing, Vol. 18. DOI: 10.1007/s11554-020-00987-8.
18. **Garcia-Lamont, F., Cervantes, J., López, A., Rodriguez, L. (2018).** Segmentation of images by color features: A survey. Neurocomputing, Vol. 292, pp. 1–27. DOI: 10.1016/j.neucom.2018.01.091.
19. **Gutierrez, A., Ansuategi, A., Susperregi, L., Tubío, C., Rankić, I., Lenža, L. (2019).** A benchmarking of learning strategies for pest detection and identification on tomato plants for autonomous scouting robots using internal databases. Journal of Sensors, Vol. 2019, pp. 1–15. DOI: 10.1155/2019/5219471.
20. **Haralick, R. M., Shanmugam, K., Dinstein, I. H. (1973).** Textural features for image classification. IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-3, No. 6, pp. 610–621. DOI: 10.1109/tsmc.1973.4309314.
21. **Hu, M. K. (1962).** Visual pattern recognition by moment invariants. IRE Transactions on Information Theory, Vol. 8, No. 2, pp. 179–187. DOI: 10.1109/tit.1962.1057692.
22. **Huang, R., Sang, N., Luo, D., Tang, Q. (2011).** Image segmentation via coherent clustering in L∗a∗b∗ color space. Pattern Recognition Letters, Vol. 32, No. 7,

pp. 891–902. DOI: 10.1016/j.patrec.2011.01.013.

23. **INEGI (2019).** Encuesta nacional agropecuaria 2019. Report ENA. 2019, Instituto Nacional de Estadística y Geografía, www.inegi.org.mx/temas/agricultura/.
24. **Jalili, L. D., Morales, A., Cervantes, J., Ruiz-Castilla, J. S. (2016).** Improving the performance of leaves identification by features selection with genetic algorithms. Workshop on engineering applications, Springer International Publishing, pp. 103–114. DOI: 10.1007/978-3-319-50880-1_10.
25. **Jiao, Z., Zhang, L., Yuan, C. A., Qin, X., Shang, L. (2019).** Plant leaf recognition based on conditional generative adversarial nets. Intelligent Computing Theories and Application, Springer International Publishing, pp. 312–319. DOI: 10.1007/978-3-030-26763-6_30.
26. **Mingqiang, Y., Kidiyo, K., Joseph, R. (2008).** A survey of shape feature extraction techniques. Pattern Recognition Techniques, Technology and Applications, Vol. 15, No. 7, pp. 43–90. DOI: 10.5772/6237.
27. **Miyao, G. (2016).** Tomato: UC IPM pest management guidelines. UC ANR Publication 3470. https://www2.ipm.ucanr.edu/agriculture/tomato/.
28. **Mokhtar, U., Ali, M. A., Hassenian, A. E., Hefny, H. (2015).** Tomato leaves diseases detection approach based on support vector machines. 2015 11th International Computer Engineering Conference (ICENCO), IEEE, pp. 246–250. DOI: 10.1109/icenco.2015.7416356.
29. **Prasad, S., Kumar, P., Hazra, R., Kumar, A. (2012).** Plant leaf disease detection using gabor wavelet transform. Swarm, Evolutionary, and Memetic Computing, Springer Berlin Heidelberg, pp. 372–379. DOI: 10.1007/978-3-642-35380-2_44.
30. **Raza, S. E. A., Prince, G., Clarkson, J. P., Rajpoot, N. M. (2015).** Automatic detection of diseased tomato plants using thermal and stereo visible light images. PLOS ONE, Vol. 10, No. 4, pp. e0123262. DOI: 10.1371/journal.pone.0123262.
31. **Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986).** Learning representations by back-propagating errors. Nature, Vol. 323, No. 6088, pp. 533–536. DOI: 10.1038/323533a0.
32. **Schor, N., Bechar, A., Ignat, T., Dombrovsky, A., Elad, Y., Berman, S. (2016).** Robotic disease detection in greenhouses: Combined detection of powdery mildew and tomato spotted wilt virus. IEEE Robotics and Automation Letters, Vol. 1, No. 1, pp. 354–360. DOI: 10.1109/lra.2016.2518214.
33. **Shijie, J., Peiyi, J., Siping, H. (2017).** Automatic detection of tomato diseases and pests based on leaf images. 2017 Chinese Automation Congress (CAC), IEEE, pp. 2537–2510. DOI: 10.1109/cac.2017.8243388.
34. **Suryawati, E., Sustika, R., Yuwana, R. S., Subekti, A., Pardede, H. F. (2018).** Deep structured convolutional neural network for tomato diseases detection. 2018 International Conference on Advanced Computer Science and Information Systems (ICACSIS), IEEE, pp. 385–390. DOI: 10.1109/icacsis.2018.8618169.
35. **Vapnik, V. N. (1999).** An overview of statistical learning theory. IEEE Transactions on Neural Networks, Vol. 10, No. 5, pp. 988–999. DOI: 10.1109/72.788640.
36. **Wang, X., Wang, Z., Zhang, S. W. (2019).** Segmenting crop disease leaf image by modified fully-convolutional networks. Intelligent Computing Theories and Application, Springer International Publishing, pp. 646–652. DOI: 10.1007/978-3-030-26763-6_62.
37. **Wu, Q., Chen, Y., Meng, J. (2020).** DCGAN-based data augmentation for tomato leaf disease identification. IEEE Access,

Vol. 8, pp. 98716–98728. DOI: 10.1109/access.2020.2997001.

38. **Zhang, K., Wu, Q., Liu, A., Meng, X. (2018).** Can deep learning identify tomato leaf disease?. Advances in Multimedia, Vol. 2018, pp. 1–10. DOI: 10.1155/2018/6710865.
39. **Zhang, Y., Song, C., Zhang, D. (2020).** Deep learning-based object detection improvement for tomato disease. IEEE Access, Vol. 8, pp. 56607–56614. DOI: 10.1109/access.2020.2982456.
40. **Zheng, Y., Yuan, C. A., Shang, L., Huang, Z. K. (2019).** Leaf recognition based on capsule network. Intelligent Computing Theories and Application, Springer International Publishing, pp. 320–325. DOI: 10.1007/978-3-030-26763-6_31.

Article received on 03/04/2021; accepted on 23/04/2024.
**Corresponding author is Ernesto García-Amaro.*

ISSN 2007-9737

Editorial for Thematic Section:

Theory and Applications of Fuzzy Systems, Neural Networks, and Metaheuristics

This thematic section deals with recent developments on fuzzy logic, neural networks and meta-heuristic optimization algorithms, as well as their hybrid combinations, and their application in areas such as, intelligent control and robotics, pattern recognition, medical diagnosis, time series prediction and optimization of complex problems.

The thematic section comprises papers dealing with type-1 and type-2 fuzzy logic, which basically consists of papers that propose new concepts and algorithms based on type-1 and type-2 fuzzy logic and their applications. We also consider papers that present theory and practice of meta-heuristics in different areas of application. Metaheuristics of interest will include genetic algorithms, particle swarm optimization, grey wolf optimization, and other recent nature-inspired optimization algorithms. In addition, special interest is given to papers presenting theory and practice of neural networks in different areas of application, including convolutional and deep learning neural networks.

In the current literature, we can find interesting papers on diverse applications of fuzzy logic, neural networks and hybrid intelligent systems in medical applications.

In addition, we can find papers describing applications of fuzzy logic, neural networks and meta-heuristics in robotics problems, pattern recognition systems, time series prediction and other areas.

This will warrant more and more research attention from the scientific community on these important topics, especially since everyday newer and newer systems are emerging across all the domains of science and engineering, e.g. social networks, big data analytics, cyber security, cyber-physical systems, cloud computing etc.

After the reviewing process, 12 papers were selected for inclusion in the thematic section.

The first paper by Miguel A. García-Morales et al. is presenting a study on the Multi-objective Evolutionary Algorithm Based on Decomposition with Adaptive Adjustment of Control Parameters to Solve the Bi-objective Internet Shopping Optimization Problem. Simulation results show the effectiveness of the proposed approach.

The second paper by Jessica González-San-Martín et al. is presenting an Advancing Cloud Task Scheduling: Recent Developments and Comparative Insights. Experimental results demonstrate the advantages of the proposed method.

The third paper by José Alfredo Brambila-Hernández et al. is presenting a Novel Dynamic Decomposition-Based Multi-objective Evolutionary Algorithm Using Reinforcement Learning Adaptive Operator Selection.

The fourth paper by Lucero Ortiz-Aguilar et al. describes a Design of Routes for Collaborative Robots in the Automobile Painting Process through a Comparison of Perturbative Heuristics for Iterated Local Search.

The fifth paper by Jesus A. Rodríguez-Arellano et al. describes an approach for Prescribed-Time Trajectory Tracking Control of Wheeled Mobile Robots Using Neural Networks and Robust Control Techniques.

The sixth paper by Andres Espinal et al. describes Grammatical Evolution with codons selection order as Intensification process.

In the seventh paper by Rodrigo Cordero-Martínez et al., an Adjustment of Convolutional and Hidden Layers using Type-1 Fuzzy Logic Applied to Diabetic Retinopathy Classification is presented.

The eighth paper by Martha Pulido et al. deals with a Bird Swarm Algorithm and Particle Swarm Optimization in Ensemble Recurrent Neural Networks Optimization for Time Series Prediction.

The ninth paper by Valentin Calzada-Ledesma et al. describes Water Stress Challenges: Mathematical Modeling of Water Resource Management.

The tenth paper by Daniel Ruelas et al. describes the Prediction of Enterprise Financial Health using Machine Learning and Financial Reasons for Taiwan Economic Companies.

The eleventh paper by Hector Martinez et al. describes a Dragonfly Algorithm with fuzzy parameter adaptation for benchmark mathematical function optimization.

Finally, the twelfth paper by Marylu Lagunes et al. describes a Comparative Study of Gorilla Troops Optimizer and Stochastic Fractal Search with fuzzy dynamic parameter adaptation.

We believe that these papers will be an important contribution to the state of the art of fuzzy systems, neural networks and metaheuristics for solving real-world problems.

In addition, there are also contributions on the theoretical side with new concepts and models of fuzzy theory, neural models and new proposed metaheuristic methods.

We envision that the papers of the thematic section will be of great interest to researchers and students of the computational intelligence areas, as well as in medical and healthcare areas.

Guest editor:

Oscar Castillo
(Tijuana Institute of Technology, TecNM, Mexico,
ocastillo@tectijuana.mx)

ISSN 2007-9737

Multi-Objective Evolutionary Algorithm based on Decomposition with Adaptive Adjustment of Control Parameters to Solve the Bi-Objective Internet Shopping Optimization Problem (MOEA/D-AACPBIShOP)

Miguel A. García-Morales[1], José A. Brambila-Hernández[1], Héctor J. Fraire-Huacuja[1,*], Juan Frausto-Solis[1], Laura Cruz-Reyes[1], Claudia G. Gómez-Santillan[1], Juan M. Carpio-Valadez[2]

[1] National Technological Institute of Mexico, Technological Institute of Madero City, Mexico

[2] National Technological Institute of Mexico, Technological Institute of Leon, Mexico

{soporteclusterlanti, hector.fraire2014, juan.frausto, cruzreyeslaura}@gmail.com, alfredo.brambila@outlook.com, claudia.gs@cdmadero.tecnm.mx, juanmartin.carpio@leon.tecnm.mx

Abstract. The main contribution of this paper is the implementation of a multi-objective evolutionary algorithm based on decomposition with adaptive adjustment of control parameters applied to the bi-objective problem of Internet shopping (MOEA/D-AACPBIShOP). For this variant of the IShOP, the minimization of the cost and shipping time of the shopping list is considered. The proposed MOEA/D-AACPBIShOP algorithm produces an approximate Pareto set on a total of nine of instances with real-world data classified as small, medium, and large. The instances are obtained using the Web Scraping technique, extracting some information attributes of technological products from the Amazon site. This optimization problem is a very little studied variant of the Internet Shopping Problem (IShOP). The proposed algorithm is compared with two multi-objective algorithms: A Non-dominated Sorting Genetic Algorithm II (NSGA-II) and the basic MOEA/D version. The results demonstrate that the three algorithms studied have a similar statistical performance with respect to the quality of the solutions they provide. To make a comparison, these algorithms are evaluated using three metrics: Hypervolume, Generalized Dispersion, and Inverted Generational Distance. On the other hand, the Wilcoxon and Friedman non-parametric tests validate the obtained results with a 5% significance level.

Keywords. Multi-objective, approximate Pareto front, evolutionary algorithm, web scraping, bi-objective.

1 Introduction

The Internet allows efficient communication throughout the world [10]. The Internet has revolutionized the way business is carried out due to the incorporation of commercial marketing, sales, and customer service tools [10].

Due to the great the importance of the Internet in organizations, E-commerce is one of the main contributors of large companies [1]. On the other hand, the Internet allows communication from multiple digital devices such as sensors, cameras, smart cities, among others [2, 3, 4]. Nowadays, this scenario is known as "The Internet Shopping Problem".

It is a classic scenario of electronic commerce due to the multiple benefits that users obtain by buying or acquiring goods or services through the Internet [5]. Online shopping makes it easier for people to access a wide variety of products and services offered by companies without having restrictions on time, place, or space [1].

In one of the most relevant works in the state-of-the-art field, the authors propose an innovative solution for the basic case of the Internet shopping problem with shipping costs.

Table 1. Notation table [10]

Variable/Parameter	Description
M	Group of stores
N	Group of products
I	Array solution
m	Number of stores, $\|M\|$
n	Number of products, $\|N\|$
j	Store indicator
i	Product indicator
N_i	Container of products available in a store j
f_j	Shipping cost of all products in the store j
p_{ij}	Cost of product i in store j
d_{ij}	Delivery time of a product i in store j
x_{ij}	Binary variable that indicates wheter producto i is purchased in store j
y_j	Binary variable indicating wheter to add the sipping cost of store j

This method consists of a memetic algorithm (MAIShOP) that incorporates standard instances, solution generation through the first-best heuristic, and a local search based on a heuristic that selects the lowest cost of each product in all stores [6].

Morales et al. [7] review the developed models, the implemented solution methods, and the instances used to analyze the performance of the algorithms described in the state-of-the-art.

Finally, it can be identified that one of the variants little investigated is the one that involves more than one optimization objective, in which the total cost of the purchase and the delivery time of products are considered.

Some Internet purchases require optimizing the total purchase cost, including the shipping cost and delivery time of different online stores [1]. Typically, users want to find the store with the lowest total cost and the shortest delivery time [1].

These decisions allow us to minimize the effort and maximize the benefit of the shopping list [10]. Chung [8] proposes a new Internet shopping optimization model that includes two objectives (total cost and delivery time) in which he incorporates for the first time a multi-objective optimization model.

Chaerani et al. [9] establishes the similarity between the model developed by Chung and the maximum flow problem with circular demand (MFP-CD) because it matches the multiples sources with respect to the multiple stores.

Chung's bi-objective model incorporates the decision variable on delivery time. Chaerani et al. [9] modifies this decision variable into an adjustable robust counterpart (ARC) method. Chaerani et al. [1] propose the Benders decomposition method to solve the Adjustable Robust Count Party Problem adapted to "the Internet Shopping Problem (ARC-ISOP)".

García-Morales et al. [10] propose a "MOEA/D algorithm to solve the bi-objective Internet shopping optimization problem (MOEA/D-BIShOP)"; this algorithm presents a basic MOEA/D version and has a clear superiority in two of the three metrics that were evaluated concerning the results of the state-of-the-art.

This research work proposes the implementation of a multi-objective evolutionary algorithm based on decomposition with adaptive adjustment of control parameters as a solution method to "the Bi-objective problem of Internet Shopping (MOEA/D-AACPBIShOP)".

In the computational feasibility tests, nine instances generated using the Web Scraping technique with data from technological products extracted from Amazon were used [10].

1.1 Definition of the Problem

This model is first proposed by Chung [8] to solve "the bi-objective Internet shopping optimization problem". "In this problem, a customer wants to buy a set of n products N online, which can be purchased in a set of m available stores M.

Algorithm 1. NSGA-II/BIShOP Algorithm

Input: cs: chromosome size, nt: number of targets, mni: maximum number of iterations, ps: population size, pc: crossing percentage, nci: number of crossed individuals, pm: mutation percentage, nmi: number of mutated individuals, m: number of stores, n: number of products, p_{ij}: price of each product, f_j: shipping cost, d_{ij}: delivery time.

Output: Pop

1: Initialize parameters: chromosome size $\boldsymbol{cs}$, number of targets $\boldsymbol{nt}$, maximum number of iterations $\boldsymbol{mni}$, population size $\boldsymbol{ps}$, crossover percentage $\boldsymbol{pc}$, number of crossed individuals $\boldsymbol{nci}$, mutation percentage $\boldsymbol{pm}$, number of mutated individuals $\boldsymbol{nmi}$.
2: $Pop \leftarrow initial\_population\ (ps)$
3: $F \leftarrow non\_dominated\_sort\ (Pop)$
4: $Pop \leftarrow crowding\_distance\ (Pop, F)$
5: $Pop \leftarrow sort\_by\_crowding\_distance\_and\_Front\ (Pop, F)$
6: $PopC \leftarrow selectionBinaryTournament\ (Pop)$
7: **while** *non stop criterion* **do**
8: $PopC \leftarrow crossover\ (PopC)$
9: $PopM \leftarrow mutation\ (PopC)$
10: $Pop \leftarrow merge\_list\ (Pop, PopC, PopM)$
11: $F \leftarrow non\_dominated\_sort\ (Pop)$
12: $Pop \leftarrow crowding\_distance\ (Pop, F)$
13: $Pop \leftarrow sort\_by\_crowding\_distance\_and\_Front\ (Pop, F)$
14: $Pop \leftarrow truncate\_list\ (Pop, ps)$
15: $F \leftarrow non\_dominated\_sort\ (Pop)$
16: $Pop \leftarrow crowding\_distance\ (Pop, F)$
17: $Pop \leftarrow sort\_by\_crowding\_distance\_and\_Front\ (Pop, F)$
18: **end while**
19: **return** Pop

Now, the set $\mathrm{N_i}$ contains the products available in store i, each product $j \in N_i$ has a cost of p_{ij}, a shipping cost f_j, and a delivery time d_{ij}. The shipping cost is charged if one or more products are purchased in the store i.

The Bi-objective Internet Shopping Optimization Problem (BIShOP) consists of minimizing the total cost of purchasing all products N, considering the cost-plus shipping costs, and minimizing the delivery time" [10]. Table 1 describes the parameters and variables used in the model.

The model presents the optimization of two objectives: one is the purchase cost, and the other is the delivery time limitation. The first objective seeks to minimize the purchase cost; the second objective seeks to minimize the delivery time of the products (see Equation 1):

$$\text{Min} \sum_i \sum_j p_{ij} x_{ij} + \sum_j f_j y_j, \quad \text{Min Max}_{i,j(d_{ij} x_{ij})}, \tag{1}$$

s.t.

$$\sum_j x_{ij} = 1, \forall i = 1, \ldots, n,$$

$$\sum_i x_{ij} \leq n y_j, j = 1, \ldots, m,$$

$$x_{ij} = 0/1, y_j = 0/1,$$

where m represents the number of stores, n the number of products, $\sum_j x_{ij} = 1$ is a limitation that indicates that the items to be purchased must be chosen only from available stores. $\sum_i x_{ij} \leq n y_j$ is a constraint that implies that a standard shipping cost will be applied every time a purchase is made in the store, regardless of the products selected, and $x_{ij} = 0/1, y_j = 0/1$ indicates that decision variables can only take binary values.

2 General Structure of Multi Objective Algorithms Applied to BIShOP

This section provides a detailed explanation of the essential components that form the multi-objective optimization algorithms utilized in "BIShOP". To represent each solution in the population, these algorithms employ a vector representation, which is an I vector of N length. This I vector includes all the stores from where the products can be purchased. Equation 1 shows in detail how the calculation of the objective functions is carried out.

2.1 Crossover Operator

This operator randomly selects two solutions called parent_1 and parent_2 [11]. The solution child_1 is generated by taking the initial half of parent_1 and joining it with the second half of parent_2. Later, to form child_2, the initial half of parent_2 is joined with the second half of parent_1 [12]. Subsequently, a random number is generated; if this generated value is less than 0.5, the crossover operator selects child_1; otherwise, it takes child_2 to advance to the mutation process.

The crossover operator uses $\lfloor N/2 \rfloor$ or $\lceil N/2 \rceil$ as the crossover point. In the case of the MOEA/D algorithm, the crossover operator selects only one of the generated children and randomly decides

which one will continue with the mutation process [10]". The NSGA-II algorithm allows both offspring generated during the crossover process to advance to the mutation process.

2.2 Mutation Operator

The mutation process of the MOEA/D algorithm takes the candidate solution selected by the crossover operator. It immediately positions itself on the first element of the solution and generates a random number; if this random value is less μ, the current element of the solution is replaced by a random value in the online stores range $[1, m]$ [10].

This process continues until all elements of the current solution have been examined". The mutation process of the NSGA-II algorithm goes through all the elements of the vector and searches in which store that product has the lowest cost. This search ends when all stores in all products have been reviewed.

2.3 The Non-Dominated Sorting Genetic Algorithm II to Solve the IShOP Bi-Objective Problem (NSGA-II/BIShOP)

NSGA-II is a multi-objective optimization algorithm proposed as an improvement of NSGA [13], it uses the structure of genetic algorithms and is based on these principles: the best individuals never disappear from the population and during the selection if two non-dominated solutions are found, the most diverse one is preferred.

Algorithm 1 describes the general structure of the NSGA-II algorithm applied to the BIShOP problem. The algorithm in step 1 starts by defining the parameters such as chromosome size cs, number of targets nt, maximum number of iterations mni, population size ps, crossover percentage pc, number of crossed individuals nci, mutation percentage pm and number of mutated individuals nmi.

In step 2, a Pop population is created randomly. From steps 3 to 5, the population is ordered according to the levels of non-dominance (ordering of the Pareto fronts: F_1, F_2,…). Each solution is assigned a fitness function according to its level of non-dominance (1 is the best level) and it is understood that this function must be decreased during the process. In step 6, binary tournament selection is applied, and a new population called PopC is obtained.

The population obtained in the previous step is used in the crossover operator and is updated in step 8. In step 9, the mutation operator is applied and a new population of PopM descendants is obtained. In step 10, the three populations (Pop, PopC and PopM) are joined. From steps 11 to 13, a ranking is assigned to each individual in the fronts and the crowding distance is obtained, subsequently they are ordered, first by fronts from lowest to highest and then by crowding distance from highest to lowest. In step 14 the list of elements is truncated to leave only the best individuals, and which fits the initial ps. From steps 15 to 17, the previous process is applied again, only to the population that was obtained in the previous steps. Finally, in step 19 the NSGA-II algorithm returns the front with the best individuals obtained in the entire process.

Algorithm 1 NSGA-II/BIShOP Algorithm

Input: cs: chromosome size, nt: number of targets, mni: maximum number of iterations, ps: population size, pc: crossing percentage, nci: number of crossed individuals, pm: mutation percentage, nmi: number of mutated individuals, m: number of stores, n: number of products, p_{ij}: price of each product, f_j: shipping cost, d_{ij}: delivery time.

Output: Pop

1: Initialize parameters: chromosome size $\boldsymbol{cs}$, number of targets $\boldsymbol{nt}$, maximum number of iterations $\boldsymbol{mni}$, population size $\boldsymbol{ps}$, crossover percentage $\boldsymbol{pc}$, number of crossed individuals $\boldsymbol{nci}$, mutation percentage $\boldsymbol{pm}$, number of mutated individuals $\boldsymbol{nmi}$.
2: $Pop \leftarrow initial\_population\ (ps)$
3: $F \leftarrow non\_dominated\_sort\ (Pop)$
4: $Pop \leftarrow crowding\_distance\ (Pop, F)$
5: $Pop \leftarrow sort\_by\_crowding\_distance\_and\_Front\ (Pop, F)$
6: $PopC \leftarrow selectionBinaryTournament\ (Pop)$
7: ***while*** $non\ stop\ criterion$ ***do***
8: $PopC \leftarrow crossover\ (PopC)$
9: $PopM \leftarrow mutation\ (PopC)$
10: $Pop \leftarrow merge\_list\ (Pop, PopC, PopM)$

11: $F \leftarrow non\_dominated\_sort\ (Pop)$
12: $Pop \leftarrow crowding\_distance\ (Pop, F)$
13: $Pop \leftarrow sort\_by\_crowding\_distance\_and\_Front\ (Pop, F)$
14: $Pop \leftarrow truncate\_list\ (Pop, ps)$
15: $F \leftarrow non\_dominated\_sort\ (Pop)$
16: $Pop \leftarrow crowding\_distance\ (Pop, F)$
17: $Pop \leftarrow sort\_by\_crowding\_distance\_and\_Front\ (Pop, F)$
18: ***end while***
19: ***return*** Pop

2.4 The Multi-Objective Evolutionary Algorithm based on Decomposition with Adaptive Adjustment of Control Parameters to Solve the IShOP Bi-Objective Problem (MOEA/D AACPBIShOP)

The multi-objective evolutionary algorithm based on decomposition (MOEA/D) was developed by Zhang and Li [14, 15, 16] and serves as a reliable and robust alternative for working with MOPs. Initially it makes a distribution of a set of weight vectors (λ) within the objective functional space.

Subsequently it creates a matrix of T closets vectors considering the Euclidean distance between the vectors, thus generating neighborhoods [17]". The basic version of the MOEA/D algorithm uses the Tchebycheff decomposition shown in Eq. 6:

$$\min g^{te}\ (x|\lambda^j, z^*) = \max_{1\le i\le m} \frac{1}{\lambda_i^j} |f_i(x) - z_i^*|, \quad (2)$$

The MOEA/D-AACPBIShOP algorithm is represented in Algorithm 2. In steps 1 to 4, λ reference vectors are established, neighborhoods are created using the T nearest neighbor vectors as criteria, and the ideal Z point is calculated.

The main loop runs through all individuals within the population. In step 7, two parents are chosen. These are taken from the neighborhoods created in $B(i)$. $B(i)$ is traversed, and two parents are chosen randomly; then, the crossover and mutation operators are applied to generate a single child. In the final part of the algorithm, the Z value is updated again.

The aggregation values of the two are calculated using λ reference vectors; likewise, the aggregation value of the child y^i is replaced with a simple criterion: if the child y^i has an aggregation value less than one of the parents, it is replaced; otherwise, the parent remains, and the population is not modified".

Algorithm 2 MOEA/D-AACPBIShOP Algorithm

Input: MOP – Bi-objective IShOP Problem, Pop – Population, $nPop$ – Population size, $fileSize$ – File size, Stopping criterion, *N* – the number of subproblems considered in MOEA/D-AACPBIShOP, A uniform distribution of *N* weight vectors: $\lambda^1, \dots, \lambda^N$, *T* – the number of weight vectors in the neighborhoods of each weight vector

Output: EP

Functions: $FRRMAB()$: obtains an index of an action to perform, $executeAction(actionindex)$: executes an action according to an index, $SlidingWindow\ (actionindex, improvement[i])$: Sliding window that stores the index of action to be performed and the improvement in cost, $UpgradeRewards(SlidingWindow)$: Updates the sliding window rewards.

1: $EP = \emptyset$
2: Compute the Euclidean distances between any two weight vectors and then compute the weight vectors *T* closets to each weight vector.
3: ***for*** $i \leftarrow 1$ ***to*** N ***do***
4: $B(i) = \{i_1, \dots, i_T\}$ where $\lambda^{i_1}, \dots, \lambda^{i_T}$ are T nearest weight vectors λ^i
5: ***end for***
6: Generate initial population $x^1, \dots, x^N$ randomly.
7: $FV^i = F(x^i)$
8: Initialize $z = (z_1, \dots, z_m)^N$ for the bi-objective IShOP
9: ***while*** stopping criterion not met ***do***
10: $indexaction = FRRMAB()$
11: $executeAction(indexaction)$
12: ***for*** $i \leftarrow 1$ ***to*** N ***do***
13: Randomly select two indices *k, l* from *B*(*i*), and generate a new solution y from x^k and x^l using genetic operators

14: Apply a problem-specific repair/improvement heuristic on *y* to produce *y'*
15: **for** $j \leftarrow 1$ **to** m **do**
16: **if** $z_j < f_j(y')$ **then**
17: $z_j = f_j(y')$
18: **end if**
19: **end for**
20: **foreach** index $j \in B(i)$ **do**
21: **if** $g^{te}(y'|\lambda^j, z) \leq g^{te}(x^j|\lambda^j, z)$ **then**
22: $x^j = y'$
23: $FV^j = F(y')$
24: **end if**
25: **end foreach**
26: Add to $SlidingWindow(actionindex, improvement[i])$
27: $Upgrade\_Rewards(SlidingWindow)$
28: **end for**
29: remove from EP all solutions dominated by *F*(*y'*)
30: insert *F*(*y'*) in EP if there are no solutions in EP that dominate *F*(*y'*)
31: **end while**

In this research work, the modified version of the adaptive operator selection method is used to achieve adaptive adjustment of control parameters. Using the Fitness-Rate-Rank-Based Multi-armed Bandit Adaptive (FRRMAB) method [18]. The FRRMAB method avoids this problem using fitness improvement rates (FIR). The formula for calculating these rates is shown in Equation 3:

$$FIR_{i,t} = \frac{pf_{i,t} - cf_{i,t}}{pf_{i,t}}, \tag{3}$$

where $pf_{i,t}$ is the fitness value of the parent, and $cf_{i,t}$ is the fitness value of the children. The reward (Reward_i) of the actions is calculated by adding the FIR values of each action within the sliding window, they are ordered in descending order and classified, using the rank (Rank_i) for each action i [18]. In the end, only the best stocks are selected, considering the decay factor $D \in [0,1]$. Rewards Reward_i are transformed using Equation 4:

$$\text{Decay}_i = D^{\text{Rank}_i} \times \text{Reward}_i. \tag{4}$$

To assign credits to action i, use Equation 5:

$$FRR_{i,t} = \frac{\text{Decay}_i}{\sum_{j=1}^{K} \text{Decay}_j}. \tag{5}$$

The lower the D decay value, the more likely it is to influence the stock's upside. The credit allocation process is represented in Algorithm 3 [17].

Algorithm 3 Assignment of credits

1: Initialize each reward $Reward_i = 0$
2: Initialize $n_i = 0$;
3: **for** $i \leftarrow 1$ **to** SlidingWindow.length **do**
4: $action =$ SlidingWindow.GetIndexaction(i)
5: $FIR =$ SlidingWindow.GetFIR(i)
6: $Reward_{action} = Reward_{action} + FIR$
7: $n_{action} + +$
8: **endfor**
9: Rank $Reward_i$ in descending order and set $Rank_i$ t be the rank value of action i
10: **for** $action \leftarrow$ **to** K **do**
11: $Decay_{action} = D^{Rank_{action}} \times Reward_{action}$
12: **endfor**
13: $DecaySum = \sum_{action=1}^{K} Decay_{action}$
14: **for** $action \leftarrow$ **to** K **do**
15: $FRR_{action} = Decay_{action}/DecaySum$
16: **endfor**

Bandit-based action selection chooses a stock considering the credits assigned to it and using the FRR values as a quality indicator [18], this process is shown in Algorithm 4.

Algorithm 4 Bandit-based action selection

if there are actions that have not been selected **then**
action$_t$ = randomly select a security from the action pool
else

$$\text{action}_t = \text{argmax}_{i=1,\ldots,K}\left(FRR_{i,t} + C \times \sqrt{\frac{2 \times \ln\left(\sum_{j=1}^{K}\{n_{j,t}\}\right)}{n_{i,t}}}\right)$$

end if

Algorithm 5 contains the various actions that are executed when the variable $actionindex$ is evaluated and said action determines the increase

or decrease in the value of the parameters that are adjusted adaptively.

Algorithm 5 executeAction

Input: $actionindex$: value of the stock selected by the FRRMAB method.

```
Switch(actionindex)
Case 1:
    pc = pc + 0.0001;
    pm = pm + 0.0001;
    sigma = sigma + 0.0001;
break;
Case 2:
    pc = pc − 0.0001;
    pm = pm − 0.0001;
    sigma = sigma − 0.0001;
break;
Case 3:
    pc = pc + 0.0001;
break;
Case 4:
    pm = pm + 0.0001;
break;
Case 5:
    sigma = sigma + 0.0001;
break;
Case 6:
    pc = pc − 0.0001;
break;
Case 7:
    pm = pm − 0.0001;
break;
Case 8:
    sigma = sigma − 0.0001;
break;
Case 9:
    pm = pm + 0.0001;
    sigma = sigma + 0.0001;
break;
Case 10:
    pc = pc + 0.0001;
    pm = pm + 0.0001;
break;
Case 11:
    sigma = pc + 0.0001;
    sigma = sigma + 0.0001;
break;
Case 12:
    pm = pm − 0.0001;
    sigma = sigma − 0.0001;
break;
Case 13:
    pc = pc − 0.0001;
    pm = pm − 0.0001;
break;
Case 14:
    pc = pc − 0.0001;
    sigma = sigma − 0.0001;
break;
```

3 Computational Experiments

The names of instances determine their size, m is the number of stores, and n is the number of products. For the experimental test, three sets of real-world instances of different sizes were used, and each subset contains 30 instances, as can be seen in Table 2 [10]".

The designs are obtained from Web Scraping of multiple technological products (USB flash, Modem, RAM) that were carried out on Amazon's e-commerce website. In this process, approximately 8002 records containing product names, prices, suppliers, delivery time, and shipping costs were obtained [10].

Fig. 1 shows the process of building the instances from real-world data described below: collect product and store information from the Amazon.com page.

Build an application in the Python language that allows us to explore within the search engine and obtain information using the Web Scraping technique, using various keywords such as laptop, headphones, and speakers, among others.

With a depth of 10 pages for each, the Beautiful Soup Python library is used to process the information [10]. A first version of the instances has been generated, and its construction is carried out by taking the products obtained with a defined price range and the stores are obtained.

Shipping times are defined arbitrarily (randomly) with values between 1 and 5 days. For the shipping cost, four arbitrary values are used, which are assigned randomly. These values are 88, 99, 120, and 140. The types of instances generated are shown in Table 2 [10]".

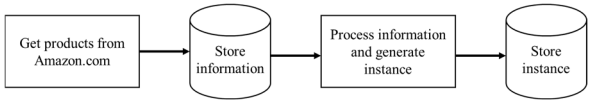


Fig. 1. General instance generation process [10]

Table 2. Definition of instances

Small	Medium	Large
3n20m	5n240m	50n400m
4n20m	5n400m	100n240m
5n20m	50n240m	100n400m

Table 3. Parameter configuration of multi-objective algorithms

Variable	MOEA/D-BIShOP	MOEA/D-AACPBIShOP	NSGA-II/BIShOP
pop	100	100	100
pc	0.5	*0.5	0.5
pm	0.01	*0.01	0.01
maxIter	100	100	100
μ	--	0.02	0.02

*Initial values before adaptive adjustment

3.1 Configuration of the Parameters

The configuration parameters of the proposed MOEA/D-AACPBIShOP algorithm is shown below pop $= 100$, pc $= 0.5$, pm $= 0.01$, maxIter $= 100$, and $\mu = 0.02$.

The above configuration was determined based on related works found in the state-of-the-art. Modifying the values of the parameters can affect the behavior of the algorithm and, therefore, the quality of the solutions.

The size of the population is important because it affects the diversity and convergence of the algorithm. A small population can lead to loss of performance, diversity, and early convergence.

An inadequate number of generations can cause the algorithm to converge prematurely or have excessive resource consumption, and incorrect use of the crossover and mutation operators can lead to deadlocks or inefficient explorations of the solution space and the size of the neighborhood because it determines the number of neighboring solutions to explore contributes to the quality of the generated solutions.

In the computational experiments, the 30 non-dominated fronts were obtained from each of the three sets of instances for each subset; subsequently, non-parametric tests were applied, and the $p - value$ was obtained to determine if there were significant differences in favor of the implemented algorithm".

Table 3 shows the parameters used for each algorithm used. The algorithms were implemented in the Java language.

3.2 Results

Tables 4, 6, and 8 organize the experimental results by metric. Friedman and Wilcoxon non-parametric tests were used with a significance level of 5%.

The first column of each table corresponds to the evaluated instance name. The second column corresponds to the reference algorithm results (MOEA/D-BIShOP). The third column contains the results of the proposed MOEA/D-AACPBIShOP and the fourth the results of the NSGA-II algorithm.

In the table, the symbol ▲ represents the statistical significance in favor of the reference algorithm, the symbol ▼ indicates that there is significant statistical difference in favor of the comparison algorithm (current column), and the symbol == means that the algorithms being compared have the same statistical performance.

The cells marked in dark gray represent the winning algorithm in a given problem and the front, and second places are marked in light gray.

3.1.1 Hypervolume

"The hypervolume (HV) calculates the volume of the objective space weakly dominated by an approximation set [17]. The first column in Table 4 represents the reference algorithm". As can be seen, in the hypervolume metric, the NSGA-II/BIShOP algorithm is better in five of the nine problems compared to the reference algorithm and compared to the MOEA/D-AACPBIShOP Algorithm it has a similar performance.

3.1.1.1 Friedman Test

"The p-value calculated with the Friedman test is 0.12110333239233029, so with a level of statistical

Table 4. Results HV (median and IQR values)

Problem	MOEA/D-BIShOP	MOEA/D-AACPBIShOP	NSGA-II/BIShOP
3n20m	0.00e+00 3.33e-01	3.33e-01 3.33e-01 ==	1.00e+00 3.33e-16▼
4n20m	0.00e+00 2.50e-01	0.00e+00 2.50e-01 ==	1.00e+00 2.50e-01▼
5n20m	0.00e+00 3.24e-01	0.00e+00 3.33e-01 ==	1.00e+00 3.33e-16▼
5n240m	0.00e+00 3.33e-01	0.00e+00 3.33e-01 ==	1.00e+00 3.33e-16▼
5n400m	0.00e+00 0.00e+00	0.00e+00 3.33e-01 ==	1.00e+00 3.33e-16▼
50n240m	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==
50n400m	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==
100n240m	1.00e+00 0.00e+00	1.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==
100n400m	1.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	1.00e+00 0.00e+00 ==

Table 5. Average ranks for HV

Algorithm	AVG Rank
NSGA-II/BIShOP	1.44
MOEA/D-AACPBIShOP	2.22
MOEA/D-BIShOP	2.33

Table 6. Results GS (Median and IQR values)

Problem	MOEA/D-BIShOP	MOEA/D-AACPBIShOP	NSGA-II/BIShOP
3n20m	4.86e-01 1.02e-01	4.83e-01 6.74e-02 ▼	4.66e-01 5.52e-02 ▼
4n20m	4.15e-01 6.55e-02	4.15e-01 6.64e-02 ==	4.05e-01 5.43e-02 ▼
5n20m	4.86e-01 1.28e-01	4.98e-01 9.84e-02 ▲	4.66e-01 5.52e-02 ▼
5n240m	4.94e-01 9.88e-02	4.89e-01 7.74e-02 ▼	4.76e-01 8.29e-02 ▼
5n400m	4.15e-01 8.92e-02	4.11e-01 7.67e-02 ▼	4.07e-01 6.41e-02 ▼
50n240m	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==
50n400m	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==
100n240m	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==
100n400m	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==	0.00e+00 0.00e+00 ==

significance of 5%, it is significant. Table 5 below shows the average ranks per algorithm obtained with the Friedman test". The Friedman test suggests that no algorithm differs significantly.

The above shows that the algorithm obtains better approximate Pareto fronts for all the evaluated instances.

3.1.2 Generalized Spread

"Generalized Spread (GS) evaluates the degree of dispersion and uniformity of the solutions identified. In Table 6, the first column is the reference algorithm".

As can be seen, in the generalized spread metric, the reference algorithm is statistically better in one of nine problems compared to the MOEA/D-AACPBIShOP Algorithm and compared to the NSGA-II/BIShOP it has a lower performance.

3.1.2.1 Friedman Test

"The p-value calculated with the Friedman test is 0.09697196786440554, so with a level of statistical significance of 5%, it is significant. Table 7 below shows the average ranks per algorithm obtained with the Friedman test". The Friedman test suggests that no algorithm differs significantly. Therefore, the approximate Pareto fronts obtained in the three algorithms have similar performance.

Table 7. Average ranks for GS

Algorithm	AVG Rank
NSGA-II/BIShOP	1.44
MOEA/D-AACPBIShOP	2.11
MOEA/D-BIShOP	2.44

Table 8. Results IGD (median and IQR values)

Problem	MOEA/D-BIShOP	MOEA/D-AACPBIShOP	NSGA-II/BIShOP
3n20m	6.94e-01 1.93e-01	6.94e-01 2.20e-01 ==	6.65e-01 2.57e-01▼
4n20m	1.58e+00 8.64e-01	1.59e+00 8.74e-01 ▲	1.53e+00 9.49e-01▼
5n20m	9.53e-01 6.41e-01	9.60e-01 6.45e-01 ▲	8.92e-01 7.43e-01▼
5n240m	1.87e+00 1.25e+00	1.89e+00 1.16e+00 ▲	1.83e+00 1.30e+00▼
5n400m	1.77e+00 1.23e+00	1.79e+00 1.12e+00 ▲	1.73e+00 1.39e+00▼
50n240m	1.34e+154 0.00e+00	1.34e+154 0.00e+00 ==	1.34e+154 0.00e+00 ==
50n400m	1.34e+154 0.00e+00	1.34e+154 0.00e+00 ==	1.34e+154 0.00e+00 ==
100n240m	1.34e+154 0.00e+00	1.34e+154 0.00e+00 ==	1.34e+154 0.00e+00 ==
100n400m	1.34e+154 0.00e+00	1.34e+154 0.00e+00 ==	1.34e+154 0.00e+00 ==

Table 9. Average ranks for IGD

Algorithm	AVG Rank
NSGA-II/BIShOP	2
MOEA/D-AACPBIShOP	2
MOEA/D-BIShOP	2

3.1.3 Inverted Generational Distance

"The inverted generation distance (IGD) gives the average distance between any point in the reference set and its nearest point in the approximation set [18]. In Table 8, the second column is considered as the reference algorithm".

As can be seen, in the generalized spread metric, the reference algorithm is statistically better in four of nine problems compared to the MOEA/D-AACPBIShOP Algorithm and compared to the NSGA-II/BIShOP it has a lower performance.

3.1.3.1 Friedman Test

"The p-value calculated with the Friedman test is 1.0, so with a level of statistical significance of 5%, it is not significant. Table 9 below shows the average ranks per algorithm obtained with the Friedman test". The Friedman test suggests that no algorithm differs significantly. Therefore, the inverted generation distance metric indicates that the three algorithms find the best solution in fewer iterations.

4 Conclusions and Future Work

Finally, with the results obtained, it is observed that the three proposed multi-objective algorithms have a statistically similar performance in three evaluated metrics, which suggests that the algorithms have good dispersion in the solutions and a similar convergence.

Therefore, it is assumed that by using other genetic operators and including new elements the performance of these new BIShOP solution methods can be improved.

This paper proposes future work to explore and develop genetic operators. They would also be very useful in online stores, Internet search engines, and other complex problems similar to BIShOP.

These tools allow Internet searches to be carried out considering more than one attribute at a time and allow more than one solution to be chosen that can provide great benefits to users and companies.

Acknowledgments

The authors thank CONAHCyT, Mexico, for the support provided through scholarships for postgraduate studies and the National System of Researchers.

References

1. **Chaerani, D., Rusyaman, E., Mahrudinda, Marcia, A., Fridayana, A. (2021).** Adjustable robust counterpart optimization model for internet shopping online problem. Journal of Physics: Conference Series, Vol. 1722, No. 1, pp. 012074. DOI: 10.1088/1742-6596/1722/1/012074.
2. **Zamir, M., Ali, N., Naseem, A., Frasteen, A. A., Zafar, B., Assam, M., Othman, M., Attia, E. (2022).** Face detection & recognition from images & videos based on CNN & raspberry pi. Computation, Vol. 10, No. 9, pp. 148. DOI: 10.3390/computation10090148.
3. **Afzal, K., Tariq, R., Aadil, F., Iqbal, Z., Ali, N., Sajid, M. (2021).** An optimized and efficient routing protocol application for IoV. Mathematical Problems in Engineering, Vol. 2021, pp. 1–32. DOI: 10.1155/2021/9977252.
4. **Malik, U. M., Javed, M. A., Zeadally, S., Islam, S. U. (2022).** Energy-efficient fog computing for 6g-enabled massive IoT: Recent trends and future opportunities. IEEE Internet of Things Journal, Vol. 9, No. 16, pp. 14572–14594. DOI: 10.1109/jiot.2021.3068056.
5. **Kumar, S. (2015).** Online shopping-a literature review. Proceedings of National Conference on Innovative Trends in Computer Science Engineering. pp. 129–131.
6. **Fraire-Huacuja, H. J., García-Morales, M. Á., López-Locés, M. C., Gómez-Santillán, C. G., Cruz-Reyes, L. Morales-Rodríguez, M. L. (2021).** Optimization of the internet shopping problem with shipping costs. Studies in Computational Intelligence, pp. 249–255. DOI: 10.1007/978-3-030-68776-2_14.
7. **García-Morales, M. Á., Fraire-Huacuja, H. J., Frausto-Solís, J., Cruz-Reyes, L., Gómez-Santillán, C. G. (2023).** A survey of models and solution methods for the internet shopping optimization problem. Studies In: Castillo, O., Melin, P. (eds) Fuzzy Logic and Neural Networks for Hybrid Intelligent System Design. Studies in Computational Intelligence, Springer, Cham., Vol. 1061, pp. 105–122. DOI: 10.1007/978-3-031-22042-5_6.
8. **Chung, J. B. (2017).** Internet shopping optimization problem with delivery constraints. Distribution Science Research, Vol. 15, No. 2, pp. 15–20.
9. **Chaerani, D., Rusyaman, E., Marcia, A., Fridayana, A. (2021).** Adjustable robust counterpart optimization model for internet shopping online problem. Journal of Physics: Conference Series, Vol. 1722, No. 1, pp. 012074. DOI: 10.1088/1742-6596/1722/1/012074.
10. **García-Morales, M. A., Brambila-Hernández, J. A., Fraire-Huacuja, H. J., Frausto-Solis, J., Cruz-Reyes, L., Gómez-Santillan, C. G., Valadez, J. M. C., Aguirre-Lam, M. A. (2024).** Multi-objective evolutionary algorithm based on decomposition to solve the bi-objective internet shopping optimization problem (MOEA/D-BIShOP). Lecture Notes in Computer Science, pp. 326–336. DOI: 10.1007/978-3-031-51940-6_24.
11. **Holland, J. H. (1975).** Adaptation in natural and artificial systems. University of Michigan Press, And Arbor, Michigan.
12. **Umbakar, A. J., Sheth, P. D. (2015).** Crossover operators in genetic algorithms: A review. ICTACT journal on soft computing, Vol. 06, No. 01, pp. 1083–1092. DOI: 10.21917/ijsc.2015.0150.
13. **Deb, K., Agrawal, S., Pratap, A., Meyarivan, T. (2000).** A fastelitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Lecture Notes in Computer Science, pp. 849–858. DOI: 10.1007/3-540-45356-3_83.
14. **Zhang, Q., Li, H. (2007).** MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation, Vol. 11, No. 6, pp. 712–731. DOI: 10.1109/tevc.2007.892759.

15. **Eiben, A., Smith, J. (2015).** Introduction to evolutionary computing. Natural Computing Series. Berlin, DOI: 10.1007/978-3-662-44874-8.
16. **Li, H., Zhang, Q. (2009).** Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. IEEE Transactions on Evolutionary Computation, Vol. 13, No. 2, pp. 284–302. DOI: 10.1109/tevc.2008.925798.
17. **García, C. (2020).** A cellular evolutionary algorithm to tackle constrained multiobjective optimization problems. Tesis de maestría, Instituto Nacional de Astrofísica, Óptica y Electrónica. Repositorio institucional del INAOE https://inaoe.repositorioinstitucional.mx/jspui/bitstream/1009/2155/1/Mc_Thesis_Cosijopii.pdf.
18. **Brambila-Hernández, J. A., García-Morales, M. Á., Fraire-Huacuja, H. J., Angel, A. B. D., Villegas-Huerta, E., Carbajal-López, R. (2023).** Experimental evaluation of adaptive operators selection methods for the dynamic multiobjective evolutionary algorithm based on decomposition (DMOEA/D). In: Castillo, O., Melin, P. (eds) Hybrid Intelligent Systems Based on Extensions of Fuzzy Logic, Neural Networks and Metaheuristics. Studies in Computational Intelligence, Vol. 1096, pp. 307—330. DOI: 10.1007/978-3-031-28999-6_20.
19. **Jiang, S., Yang, S., Yao, X., Tan, K. C., Kaiser, M., Krasnogor, N. (2018).** Benchmark problems for CEC2018 competition on dynamic multiobjective optimisation. Newcastle University.

Article received on 18/01/2024; accepted on 08/05/2024.
**Corresponding author is Héctor J. Fraire-Huacuja.*

ISSN 2007-9737

Novel Dynamic Decomposition-Based Multi-Objective Evolutionary Algorithm Using Reinforcement Learning Adaptive Operator Selection (DMOEA/D-SL)

José Alfredo Brambila-Hernández[1], Miguel Ángel García-Morales[1], Héctor Joaquín Fraire-Huacuja[1,*], Laura Cruz-Reyes[1], Claudia G. Gómez-Santillán[1], Nelson Rangel-Valdez[1], Héctor José Puga-Soberanes[2], Fausto Balderas[1]

[1] Tecnológico Nacional de México,
Instituto Tecnológico de Ciudad Madero,
Mexico

[2] Tecnológico Nacional de México,
Instituto Tecnológico de León,
Mexico

alfredo.brambila@outlook.com, {soporteclusterlanti, hector.fraire2014}@gmail.com, lauracruzreyes@itcm.edu.mx, {claudia.gs,nelson.rv, fausto.bj}@cdmadero.tecnm.mx, pugahector@yahoo.com

Abstract. Within the multi-objective (static) optimization field, various works related to the adaptive selection of genetic operators can be found. These include multi-armed bandit-based methods and probability-based methods. For dynamic multi-objective optimization, finding this type of work is very difficult. The main characteristic of dynamic multi-objective optimization is that its problems do not remain static over time; on the contrary, its objective functions and constraints change over time. Adaptive operator selection is responsible for selecting the best variation operator at a given time within a multi-objective evolutionary algorithm process. This work proposes incorporating a new adaptive operator selection method into a Dynamic Multi-objective Evolutionary Algorithm Based on Decomposition algorithm, which we call DMOEA/D-SL. This new adaptive operator selection method is based on a reinforcement learning algorithm called State-Action-Reward-State-Action Lambda or SARSA (λ). SARSA Lambda trains an Agent in an environment to make sequential decisions and learn to maximize an accumulated reward over time; in this case, select the best operator at a given moment. Eight dynamic multi-objective benchmark problems have been used to evaluate algorithm performance as test instances. Each problem produces five Pareto fronts. Three metrics were used: Inverted Generational Distance, Generalized Spread, and Hypervolume. The non-parametric statistical test of Wilcoxon was applied with a statistical significance level of 5% to validate the results.

Keywords. Adaptive, operator, selection, dynamic, multi-objective, optimization.

1 Introduction

A definition proposed by Azzouz [1] for a dynamic problem is: “dynamic multi-objective optimization problem (DMOP) is the problem of finding a vector of decision variables which satisfies a set of constraints and optimizes a vector of functions whose scalar values They represent objectives that change over time.” A DMOP can be defined as follows in Equation 1:

$$\begin{gathered}\text{Min or Max: } F(x,t) = \\ \{f_1(x,t), f_2(x,t), \dots, f_m(x,t)\} \\ x \in X^n \\ \text{s.t.} \\ g(x,t) > 0 \\ h(x,t) = 0 \\ x_i^l \leq x \leq x_i^u, i = 1,2,\dots,n,\end{gathered} \tag{1}$$

where x is the vector of decision variables, n is its size, X^n represents the solution space, f is the set of objectives to be maximized or minimized

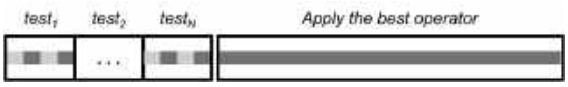


Fig. 1. Test-and-apply structure [7]

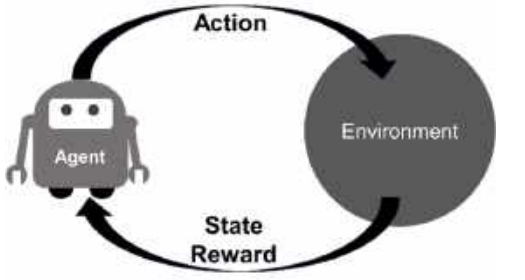


Fig. 2. Interaction Agent-Environment

concerning time, g and h represent the set of inequalities and equalities, respectively, t represents the time or dynamic nature of the problem, and m is the number of objectives.

Algorithm 1 SARSA (λ)

Parameters: α: Learning rate, ε: Policy parameter ε-greedy, γ: Discount factor, λ: Controls the amount of forward and backward learning

Variables: S_t: Current state, S_{t+1}: Next state, A_t: Current action, Reward: Reward calculated for the next state and the current action, QTable: Table of Q values representing the state-action value function

```
Initialize QTable()
foreach episode do
    S_t <- InitialState
    foreach step of episode do
        A_t ← chooseAction(S_t, ε)
        S_{t+1} ← takeAction(A_t)
        A_{t+1} ← chooseAction(S_{t+1}, ε)
        updateTraces(S_t, A_t) //e(S_t, A_t) < −λ × γ * e(S_t, A_t) + 1
        Reward ← getReward(S_{t+1}, A_t)
        Delta ← Reward + γQ(S_{t+1}, A_{t+1}) − Q(S_t, A_t)
        updateQTable(S_t, A_t, Reward, S_{t+1}, δ)
        S_t ← S_{t+1}
        A_t ← A_{t+1}
    end for
end for
```

Ke Li [2] mentions that the recurring problems within single-objective and multi-objective optimization are determining a specific configuration for the control parameters and the selection of the correct genetic operators for each type of problem; this leads us to have different scenarios for solving problems, this becomes more complicated when the person is not an EA expert.

Search operators are of great importance in metaheuristics. Some operators are better for solving a specific problem. On the other hand, the selection and order of use of these operators can affect the performance of an algorithm [3].

In this case, we focus on selecting the correct genetic operator; adaptive operator selection, also called AOS, is used. The AOS is responsible for automatically determining which variation operator to use at a given time within a MOEA process.

Within the state of the art for AOS, the most recent works found correspond to methods based on multi-arm bandits applied to static multi objective optimization algorithms based on decomposition.

In this work, a new adaptive operator selection mechanism is proposed, which is based on the SARSA (λ) reinforcement learning technique. This new AOS mechanism has been incorporated into the DMOEA/D algorithm and compared against the state-of-the-art algorithm, which was also applied to the DMOEA/D algorithm.

2 Background and Related Work

The AOS comprises two main tasks [4]: credit assignment and operator selection. In the first task, the reward or weight of each operator is determined by their performance. The second task selects the best available operator.

Credits can be assigned in different ways depending on our method or algorithm. However, in general, it can be done based on the improvement of the children's fitness concerning the parents, and it can also be done based on a ranking of the operators.

Depending on how the operator selection task works, the AOS is divided into two groups: one encompasses all methods based on a probabilistic approach, and the other encompasses a multi-arm bandit approach (MAB).

Algorithm 2. DMOEA/D-SL

Input: DMOP: Dynamic multi-objective problem,
Pop: Population List,
nPop: Population size,
fileSize: MOEA/D file size,
N: The number of subproblems considered in MOEA/D,
A: Uniform distribution of N weight vectors: $\lambda^1, \ldots, \lambda^N$,
T: The number of weight vectors in the neighborhood of each weight vector,
numStates: Number of states for algorithm SARSA (λ),
numActions: Number of actions / genetic operators,
α: Learning rate,
ε: Policy parameter ε-greedy,
γ: Discount factor,
λ: Controls the amount of forward and backward learning,
nT: Change frequency,
tauT: Severity of change

Output: Front$_f$: Pareto estimate front for each problem change

$EP = \emptyset$

Compute the Euclidean distances between any two weight vectors and then compute the closest weight vectors *T* to each weight vector.

Agent ← SarsaLambdaAlgorithm(numStates, numActions, α, γ, ε, λ)

for $i \leftarrow 1$ **to** N **do**
  $B(i) = \{i_1, \ldots, i_T\}$
  /*where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i */
end for

Generate an initial pop $x^1, \ldots, x^N$ randomly for the specific problem.
$FV^i = F(x^i)$
Initialize $z = (z_1, \ldots, z_m)^N$ for the specific problem
time=0
detectors=GetDetectorList(Pop)
it=0
f=1

while stopping criteria not met **do**
  $S_t \leftarrow$ initialState
  $A_t \leftarrow$ Agent.chooseAction(S_t)
  QVal$_t \leftarrow$ Agent.getQValue(S_t, A_t)

  for $i \leftarrow 1$ **to** $nPop$ **do**
    $y' \leftarrow$ takeAction(A_t)
    for $j \leftarrow 1$ **to** m **do**
      if $z_j < f_j(y')$ **then**
        $z_j \leftarrow f_j(y')$
      end if
    end for

    foreach index $j \in B(i)$ **do**

      if $g^{te}(y'|\lambda^j, z) \leq g^{te}(x^j|\lambda^j, z)$ **then**
        $x^j \leftarrow y'$
        $FV^j \leftarrow F(y')$
        $S_t \leftarrow S_0$
        $S_{t+1} \leftarrow$ nextState()
        $A_{t+1} \leftarrow$ Agent.chooseAction(S_{t+1})
        $QVal_{t+1} \leftarrow$ Agent.getQValue(S_{t+1}, A_{t+1})
        Agent.updateEligibilityTraces(S_t, A_t)
        reward ← Agent.getReward(S_t, A_t)
        $\delta \leftarrow$ reward + $\gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$
        Agent.updateQTable(reward, $QVal_t, QVal_{t+1}, \delta$)
        $S_t \leftarrow S_{t+1}$
        $A_t \leftarrow A_{t+1}$
        $QVal_t \leftarrow QVal_{t+1}$

      else if
        $S_t \leftarrow S_1$
        $S_{t+1} \leftarrow$ nextState()
        $A_{t+1} \leftarrow$ Agent.chooseAction(S_{t+1})
        $QVal_{t+1} \leftarrow$ Agent.getQValue(S_{t+1}, A_{t+1})
        Agent.updateEligibilityTraces(S_t, A_t)
        reward ← Agent.getReward(S_t, A_t)
        $\delta \leftarrow$ reward + $\gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$
        Agent.updateQTable(reward, $QVal_t, QVal_{t+1}, \delta$)
        $S_t \leftarrow S_{t+1}$
        $A_t \leftarrow A_{t+1}$
        $QVal_t \leftarrow QVal_{t+1}$
      end if
    end foreach
  end for

  Remove from EP all solutions dominated by $F(y')$
  Insert $F(y')$ in EP if there are no solutions in EP that dominate $F(y')$
  Time = $\frac{1}{nT} \times \left\lfloor \frac{1 \times it}{\text{tauT}} \right\rfloor$

  if problemChangeDetection(decetors, time) **then**
    insert EP into Front$_f$
    Initialize $z = (z_1, \ldots, z_m)^N$ for the specific problem
    Evualuate Pop
    Execute changeResponseMechanism()
    $FV^i = F(x^i)$
    Remove from EP all solutions dominated by $F(y')$
    Insert $F(y')$ in EP if there are no solutions in EP that dominate $F(y')$
    $f = f + 1$
  end if

  Agent.reinitializeQTable()
  $it = it + 1$
end while
return Front$_f$

In the multi-arm bandit approach, AOS uses the "multi-arm bandit (MAB) problem paradigm" [2], which considers each operator as an arm of a slot machine, each with an unknown reward probability. These methods seek to maximize the reward accumulated during the process and model these rewards to select the best operator (arm) at each moment.

In our previous work [5], we have applied "the Fitness-Rate-Rank-based Multi-Armed Bandit (FRRMAB)" [2], "Adaptive Operator Selection Based on Dynamic Thompson Sampling (DYTS)" [6], and "Adaptive operator selection with test-and-apply structure for decomposition-based multi-objective optimization (TAOS)" [7] methods to a dynamic version of the MOEA/D algorithm to observe its behavior.

Another of the most recent works in the area is "A novel bicriteria assisted adaptive operator selection (B-AOS) strategy for decomposition-based multi-objective evolutionary algorithms (MOEA/Ds)" proposed in 2021 by Wu Lin [8]. This approach employs two groups of operators; each group includes two genetic operators with different search patterns.

Table 1. Benchmark problems

Problem	Objectives	Change frequency	Change severity
dMOP1 [15]	2	100	10
dMOP2 [15]	2	100	10
FDA1 [16]	2	100	10
FDA3 [16]	2	100	10
DF4 [17]	2	100	10
DF6 [17]	2	100	10
DF10 [17]	3	100	10
DF12 [17]	3	100	10

Table 2. Algorithm parameters

Variables/ Parameters	DMOEA/D-TAOS	DMOEA/D-SL
maxIt	100	100
nPop	100	100
fileSize	100	100
Zeta	0.2	0.2
K	1	--
alpha	--	0.04
gamma	--	0.08
lambda	--	0.07
epsilon	--	0.1

In addition, it uses two criteria, which emphasize convergence and diversity, to help select the appropriate operator. In the probabilistic approach, a probability is attached to each operator, and its selection process is similar to that of a roulette wheel; the operators with the highest probabilities will be those who will have a larger area on the roulette wheel and will be the ones who will have a greater chance of being selected.

The most popular methods within this group are Probability Matching [9] and Adaptive Pursuit [10]. Another work is "Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification" which uses a probability-based AOS within a multi-objective genetic algorithm to solve feature selection problems [11].

The strategies mentioned above are applied to static multi-objective algorithms in the state-of-the-art. In this work, it has been decided to use the most recent strategy, "Adaptive operator selection with test-and-apply structure for decomposition-based multi-objective optimization (TAOS)" [7] and apply it to a dynamic multi-objective algorithm to compare the state-of-the-art strategy with the strategy proposed in this work.

2.1 Adaptive Operator Selection with Test-and-Apply Structure for Decomposition-based Multi-Objective Optimization (TAOS)

In this approach proposed by Lisha Dong in 2022 [6], the whole evolutive process is structured into several continuous sections, each designed to execute testing and application phases.

2.1.1 Test Phase

In the testing phase, each operator is tested in the same environment. The testing phase is divided into N parts, as shown in Figure 1: $test_1$, …, and $test_N$. All operators will be evaluated once in order. Therefore, the total number of function evaluations for the testing phase is defined as follows, as shown in Equation 2:

$$FE_{\text{test}} = K \times N, \tag{2}$$

where K denotes the number of operators, and the population size is defined by N. When an operator is applied in the search process, its impact needs to be measured; a successful update indicator (SUI) is introduced to assign credits to the operator. SUI is defined as follows as sampled in Equation 3:

$$SUI = \begin{cases} 1, \text{ if the generated solution is better than at least one solution in the population,} \\ 0, \text{ and otherwise.} \end{cases} \tag{3}$$

A child solution updates the first solution, and the SUI value changes from 0 to 1; if the child continues updating the solutions, the SUI will remain unchanged.

Table 3. Hypervolume (median and IQR values)

Problem	Front	DMOEA/D-TAOS	DMOEA/D-SL
FDA1	1	6.32e-01 6.24e-03	6.31e-01 8.39e-03 ▲
	2	6.33e-01 5.76e-03	6.32e-01 5.35e-03 ▲
	3	6.29e-01 7.83e-03	6.30e-01 8.70e-03 ==
	4	6.33e-01 5.44e-03	6.33e-01 8.34e-03 ▼
	5	6.31e-01 7.67e-03	6.35e-01 5.83e-03 ▼
FDA3	1	6.56e-01 5.97e-03	6.55e-01 4.65e-03 ==
	2	6.86e-01 7.86e-03	6.84e-01 4.33e-03 ==
	3	6.63e-01 3.52e-03	6.61e-01 4.50e-03 ▲
	4	6.51e-01 3.86e-03	6.49e-01 4.44e-03 ▲
	5	6.50e-01 6.20e-03	6.47e-01 8.16e-03 ▲
DMOP1	1	4.28e-01 3.71e-03	4.27e-01 4.36e-03 ▲
	2	4.07e-01 3.13e-03	4.07e-01 2.48e-03 ==
	3	3.88e-01 1.98e-03	3.88e-01 2.34e-03 ▲
	4	3.72e-01 2.27e-03	3.71e-01 2.42e-03 ==
	5	3.58e-01 2.20e-03	3.58e-01 1.96e-03 ==
DMOP2	1	4.28e-01 2.30e-03	4.27e-01 1.93e-03 ▲
	2	4.05e-01 2.61e-03	4.05e-01 3.10e-03 ▲
	3	3.86e-01 2.55e-03	3.86e-01 3.10e-03 ▼
	4	3.69e-01 2.25e-03	3.68e-01 2.53e-03 ▲
	5	3.55e-01 1.49e-03	3.55e-01 3.76e-03 ▼
DF4	1	6.83e-01 4.06e-03	6.80e-01 8.08e-03 ▲
	2	7.34e-01 4.68e-03	7.32e-01 3.60e-03 ▲
	3	7.72e-01 3.46e-03	7.72e-01 3.36e-03 ==
	4	8.04e-01 2.04e-03	8.03e-01 2.91e-03 ▲
	5	8.26e-01 3.16e-03	8.26e-01 3.39e-03 ▲
DF6	1	3.71e-02 6.53e-04	3.70e-02 9.11e-04 ▲
	2	0.00e+00 0.00e+00	0.00e+00 0.00e+00 ==
	3	3.51e-02 1.06e-01	3.81e-02 2.02e-01 ▼
	4	1.06e-01 1.75e-01	1.08e-01 2.97e-01 ▼
	5	2.12e-01 2.13e-01	2.15e-01 3.40e-01 ▼
DF10	1	8.36e-01 6.66e-03	8.36e-01 5.48e-03 ▼
	2	8.30e-01 7.12e-03	8.31e-01 5.09e-03 ▼
	3	8.11e-01 6.06e-03	8.12e-01 5.20e-03 ▼
	4	7.82e-01 5.58e-03	7.82e-01 5.14e-03 ▲
	5	7.45e-01 6.38e-03	7.45e-01 4.35e-03 ▼
DF12	1	3.04e-01 1.18e-02	3.06e-01 2.47e-02 ==
	2	6.53e-01 1.39e-02	6.47e-01 1.71e-02 ▲
	3	6.46e-01 1.16e-02	6.51e-01 1.17e-02 ==
	4	6.53e-01 1.63e-02	6.50e-01 1.90e-02 ▲
	5	6.53e-01 1.57e-02	6.46e-01 2.71e-02 ▲

2.1.1 Apply Phase

When the testing phase has finished, the successful updates count (SUC) is obtained for each operator; this is shown in Equation 4:

$$SUC_{op} = \sum_{i=1} SUI_{op}^{i}, \tag{4}$$

where SUI_{op}^{i} indicates that the operator op is tested in evaluating the ith function of the testing phase. To select the operator, we compare them with each other.

The operator with the SUC with the highest value will be the one selected to be applied in the application phase of that segment. The number of function evaluations for the application phase in each segment is defined in Equation 5:

Table 4. Generalized Spread (median and IQR values)

Problem	Front	DMOEA/D-TAOS	DMOEA/D-SL
FDA1	1	5.16e-01 1.91e-01	4.63e-01 1.90e-01 ▼
	2	5.58e-01 1.45e-01	4.38e-01 1.20e-01 ▼
	3	5.34e-01 1.67e-01	4.68e-01 1.49e-01 ▼
	4	5.27e-01 1.59e-01	4.82e-01 1.50e-01 ▼
	5	5.59e-01 1.40e-01	4.65e-01 1.29e-01 ▼
FDA3	1	6.65e-01 2.03e-01	6.96e-01 1.57e-01 ==
	2	6.22e-01 2.27e-01	5.86e-01 3.57e-01 ▼
	3	8.44e-01 1.47e-01	8.34e-01 1.41e-01 ▼
	4	6.48e-01 7.09e-02	6.37e-01 6.86e-02 ▼
	5	4.70e-01 4.49e-02	4.44e-01 4.73e-02 ▼
DMOP1	1	7.46e-01 5.48e-02	7.33e-01 1.47e-01 ==
	2	7.42e-01 7.13e-02	7.27e-01 6.02e-02 ▼
	3	7.62e-01 1.05e-01	7.45e-01 6.63e-02 ▼
	4	7.53e-01 7.05e-02	7.48e-01 7.32e-02 ▼
	5	7.40e-01 1.17e-01	7.37e-01 6.95e-02 ==
DMOP2	1	7.27e-01 7.21e-02	7.50e-01 9.14e-02 ▲
	2	6.86e-01 8.37e-02	6.73e-01 7.78e-02 ▼
	3	6.92e-01 9.39e-02	7.00e-01 7.33e-02 ==
	4	6.86e-01 5.62e-02	6.93e-01 1.12e-01 ==
	5	6.85e-01 6.99e-02	6.83e-01 6.99e-02 ==
DF4	1	8.38e-01 1.47e-01	8.77e-01 1.78e-01 ==
	2	8.39e-01 1.11e-01	8.56e-01 1.47e-01 ==
	3	1.00e+00 4.66e-02	9.83e-01 6.08e-02 ▼
	4	1.12e+00 3.52e-02	1.10e+00 5.37e-02 ▼
	5	1.16e+00 7.50e-02	1.15e+00 5.48e-02 ▼
DF6	1	7.02e-01 8.48e-02	6.96e-01 8.28e-02 ==
	2	9.41e-03 4.32e-03	9.20e-03 1.12e-02 ==
	3	1.05e-01 4.86e-02	1.10e-01 1.01e-01 ▲
	4	1.07e-01 3.28e-02	1.10e-01 7.07e-02 ▲
	5	1.29e-01 8.68e-03	1.33e-01 3.10e-02 ▲
DF10	1	7.32e-01 1.49e-01	7.61e-01 1.50e-01 ▲
	2	7.78e-01 1.78e-01	7.93e-01 1.11e-01 ==
	3	7.66e-01 1.69e-01	7.73e-01 1.15e-01 ▲
	4	7.54e-01 1.83e-01	7.50e-01 9.83e-02 ▼
	5	7.96e-01 2.02e-01	7.57e-01 1.38e-01 ==
DF12	1	5.14e-01 3.90e-02	4.76e-01 5.83e-02 ▼
	2	4.96e-01 4.41e-02	5.05e-01 3.89e-02 ▲
	3	5.14e-01 3.21e-02	5.07e-01 3.29e-02 ▼
	4	5.12e-01 5.60e-02	5.14e-01 3.97e-02 ▲
	5	5.05e-01 5.91e-02	4.91e-01 5.79e-02 ▼

$$FE_{\text{apply}} = FE_{\text{test}} \times k, \quad (5)$$

where k handles the resources for the testing and application phases, a shorter value of k, i.e., $k < 1$, assigns more resources to the testing phase versus the application phase; a larger value of k makes fewer resources available for the testing phase.

2.2 Reinforcement Learning

Reinforcement learning is “learning how to do and map situations to actions to maximize a numerical reward signal. An Agent needs to be told what actions to take; instead, she must discover which actions yield the most significant reward by attempting them” [12].

Table 5. Inverted generational distance (median and IQR values)

Problem	Front	DMOEA/D-TAOS	DMOEA/D-SL
FDA1	1	7.49e-04 3.08e-04	8.06e-04 2.87e-04 ==
	2	8.25e-04 2.43e-04	7.31e-04 3.24e-04 ==
	3	8.49e-04 2.67e-04	7.99e-04 2.93e-04 ▼
	4	7.71e-04 2.71e-04	7.14e-04 2.70e-04 ▼
	5	8.33e-04 2.82e-04	7.00e-04 2.38e-04 ▼
FDA3	1	6.80e-04 4.16e-04	6.92e-04 3.22e-04 ▲
	2	2.20e-03 1.25e-03	1.97e-03 1.46e-03 ▼
	3	2.93e-03 2.27e-03	2.90e-03 2.30e-03 ▼
	4	3.21e-03 3.31e-03	3.27e-03 2.57e-03 ▲
	5	3.78e-03 2.67e-03	4.57e-03 4.78e-03 ==
DMOP1	1	5.43e-04 1.13e-04	5.19e-04 1.96e-04 ==
	2	5.04e-04 1.43e-04	4.63e-04 9.23e-05 ▼
	3	5.29e-04 2.26e-04	4.98e-04 9.75e-05 ▼
	4	5.13e-04 1.59e-04	5.12e-04 1.40e-04 ▼
	5	5.08e-04 2.18e-04	4.92e-04 1.20e-04 ▼
DMOP2	1	5.34e-04 1.34e-04	5.33e-04 1.64e-04 ▼
	2	4.76e-04 1.54e-04	4.72e-04 1.12e-04 ==
	3	5.18e-04 1.37e-04	5.09e-04 1.26e-04 ==
	4	4.94e-04 1.13e-04	5.13e-04 1.83e-04 ▲
	5	4.79e-04 1.26e-04	4.76e-04 1.53e-04 ▼
DF4	1	2.99e-03 1.43e-03	3.57e-03 1.57e-03 ▲
	2	2.86e-03 1.01e-03	3.07e-03 1.71e-03 ▲
	3	5.49e-03 5.07e-04	5.35e-03 5.72e-04 ▼
	4	8.52e-03 3.20e-04	8.46e-03 3.85e-04 ▼
	5	1.12e-02 1.95e-04	1.12e-02 2.76e-04 ▲
DF6	1	3.86e-04 1.61e-04	4.00e-04 1.73e-04 ▲
	2	2.01e-01 1.92e-01	2.04e-01 2.74e-01 ==
	3	1.51e-02 7.50e-03	1.15e-02 1.07e-02 ▼
	4	1.43e-02 7.36e-03	1.07e-02 1.06e-02 ▼
	5	1.24e-02 6.55e-03	9.28e-03 9.45e-03 ▼
DF10	1	6.62e-03 5.77e-04	6.50e-03 4.09e-04 ▼
	2	6.48e-03 5.11e-04	6.43e-03 2.15e-04 ▼
	3	6.52e-03 5.74e-04	6.50e-03 3.29e-04 ▼
	4	6.62e-03 4.84e-04	6.66e-03 2.71e-04 ==
	5	6.76e-03 3.57e-04	6.81e-03 2.04e-04 ==
DF12	1	2.21e-03 1.43e-04	2.10e-03 3.25e-04 ▼
	2	2.17e-03 2.94e-04	2.24e-03 3.61e-04 ▲
	3	2.21e-03 2.13e-04	2.18e-03 2.05e-04 ▼
	4	1.97e-03 2.39e-04	1.98e-03 1.94e-04 ▲
	5	2.13e-03 2.49e-04	2.09e-03 1.89e-04 ==

In Figure 2, it can be seen that two important components of reinforcement learning are the Agent and the Environment. The Agent is the model that needs to be trained to make decisions.

2.2.1 SARSA (λ)

SARSA (λ) is a reinforcement learning algorithm used to train an Agent in an environment to make sequential decisions and learn to maximize a reward signal accumulated over time. It is a variant of the SARSA algorithm (State-Action-Reward-State-Action), which introduces an additional parameter called "lambda" (λ) to allow a trade-off between forward and backward learning.

The SARSA (λ) algorithm applies the TD (λ) prediction method to state-action pairs rather than

just states, requiring a trace for each pair. Let $E_t\ (s, a)$ be the trace of the state-action pair s, a [9].

Next, algorithm 1 presents the general mechanism of SARSA (λ). The eligibility trace update is defined in Equation 6:

$$e(S_t, A_t) \leftarrow \lambda\gamma e(S_t, A_t) + 1. \quad (6)$$

The calculation of the temporal error is defined in the Equation 7:

$$\delta \leftarrow \text{Reward} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t). \quad (7)$$

The temporal error is used to update the Q values and improve the estimation of the quality of the actions in the different states. The update of the Q values is defined by the Equation 8:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\delta e(S_t, A_t), \quad (8)$$

where $e(S_t, A_t)$ is the eligibility trace for the current state-action pair, δ is the temporal error, $Reward$ is the calculated reward, α is the learning rate, ε is the policy parameter $\varepsilon -$ greedy, γ is a discount factor, and λ controls the amount of forward and backward learning.

3 Proposed Algorithm DMOEA/D-SL

This work proposes a new AOS method using a reinforcement learning technique called SARSA (λ) or SARSA Lambda. Furthermore, this new AOS has been integrated into a dynamic MOEA/D algorithm (DMOEA/D).

Two mechanisms proposed by Deb [13] have been added to the MOEA/D algorithm proposed by Zhang and Li in 2007 [14] to make an algorithm capable of working with dynamic problems, a change detection mechanism based on detectors, and the change response mechanism called "A".

We have taken the multi-arm approach and used it as actions, in this case four variants of the differential evolution trader are being used as actions The SARSA Lambda mechanism has been incorporated into the main loop of DMOEA/D. The general structure of this integration is shown in algorithm 2.

SARSA Lambda is initialized with the values corresponding to each of its parameters and is assigned to an object called an Agent; in this step, QTable is also initialized (line 3). For the episode loop, the main DMOEA/D loop is used (line 14), and for the step loop, the loop that runs through the population list (line 18) has been used.

In each turn of the main loop, S_t is initialized in the initial state S_0 (line 15), and the Agent assigns A_t an action based on S_t. To select an action, it uses an ε-greedy policy, where a random number is generated between $[0, 1]$; if the generated value is less than ε, an action (operator) is taken randomly, otherwise best action for the given state is selected (line 16).

We get all the values stored in the QTable for S_t and A_t and assign them to $QVal_t$ (line 17). Once inside the loop that runs through the population, we apply the action (operator) obtained and produce a child called y' (line 19). If y' is better than any of the neighbors of individual i state S_0 is assigned to S_t; otherwise, S_t will be assigned the following state (lines 26-52).

Subsequently, the following action, A_{t+1}, is calculated based on the state S_{t+1}, the value Q_{t+1} is obtained from the QTable for S_{t+1} and A_{t+1}, the eligibility traces are updated with S_t and A_t, the reward for S_t and A_t is calculated, the delta value is calculated, and the QTable is updated. Finally, the state S_t and the current action A_t are updated.

This process is repeated until the stopping criterion has been met. It should also be considered that in each problem change, the QTable must be reinitialized (line 67).

2.2 Actions Pool

Five genetic operators are evaluated. Four different versions of the differential evolution (DE) crossover operator [6] were tested as actions. In addition to each crossover operator, the polynomial mutation operator was also applied:

Action 1: apply DE/rand/1 , $v^i \leftarrow x^i + F \times (x^{r1} - x^{r2})$,

Action 2: apply DE/rand/2 , $v^i \leftarrow x^i + F \times (x^{r1} - x^{r2}) + F \times (x^{r3} - x^{r4})$,

Action 3: apply DE/current-to-rand/1 , $v^i \leftarrow x^i + K \times \left(x^i - x^{r1}\right) + F \times (x^{r2} - x^{r3}) + F \times (x^{r4} - x^{r5})$,

Action 4: apply DE/current-to-rand/2 , $v^i \leftarrow x^i + K \times \left(x^i - x^{r1}\right) + F \times (x^{r2} - x^{r3})$.

The four crossover operators use F=0.5 and CR=1.0, and polynomial mutation with a distribution index of 20 and a 20% of mutation probability.

4 Computational Experiments

Table 1 shows the eight dynamic multi-objective benchmark problems of 2 and 3 objectives used in this experiment. For each algorithm and front of the dynamic multi-objective problem, 30 independent runs were conducted.

The objective of the experimentation is to compare the proposed algorithm against the state-of-the-art algorithm called "Adaptive operator selection with test-and-apply structure for decomposition-based multi-objective optimization (TAOS)" [6]. Table 2 shows the parameters used for each algorithm used. The algorithms were implemented in the Java language.

The values of the parameters for MOEA/D have been taken from state-of-the-art, and the values of the parameters of the SARSA Lambda Agent have been obtained by assigning values arbitrarily by performing multiple experiments to determine the current values.

4.1 Results

The experimentation results are presented below in a table by metric (hypervolume, generalized spread, and inverted generational distance). Wilcoxon non-parametric test was applied with a significance level of 5%.

The first column in the table presents the problem. The second column presents the problem front. Columns three to four present the results of each algorithm. The algorithm in the third column is taken as a reference (MOEA/D-TAOS).

The following symbols are included in the results tables: the symbol ▲ means that there is statistical significance in favor of the reference algorithm, ▼ that there is statistical significance in favor of the algorithm that is compared with the reference algorithm (in favor of the current column), and == means there is no statistical significance.

The cells marked in dark gray represent the winning algorithm in a given problem and the front, and second places are marked in light gray.

4.1.1 Hypervolume

The hypervolume denotes the multidimensional volume of the objective space weakly dominated by an approximation set [5]. In Table 3, the third column is considered as the reference algorithm. As seen in the previous table, in the hypervolume metric, with the Wilcoxon test for the DMOEA/D-SL algorithm, 8 first places are obtained with statistical significance in favor. In comparison, for the DMOEA/D-TAOS algorithm, they obtained 18 first places with statistical significance in their favor.

4.1.2 Generalized Spread

Generalized Spread measures the uniformity and their dispersion of the solutions found [5]. In Table 4, the third column is considered as the reference algorithm. As seen in the previous table, in the generalized spread metric, with the Wilcoxon test for the DMOEA/D-SL algorithm, 20 first places with statistical significance are obtained in favor. In contrast, for the DMOEA/D-TAOS algorithm, 8 first places are obtained with statistical significance in favor.

4.1.3 Inverted Generational Distance

The inverted generation distance provides the average distance between any point in the reference set and its closest point in the approximation set [5]. In Table 5, the third column is considered as the reference algorithm. As seen in the previous table, in the inverted generational distance metric, with the Wilcoxon test for the DMOEA/D-SL algorithm, 21 first places are obtained with statistical significance in favor.

In comparison, TAOS obtains 9 first places with statistical significance in favor. The results obtained for the group of problems presented in this experiment suggest that the proposed algorithm is better in generalized spread and inverted generational distance metrics.

These metrics indicate that the proposed algorithm produces quality solutions with good approximation to the Pareto front and good dispersion. Regarding the hypervolume metric, the clear winner is the state-of-the-art algorithm. The main limitation of using an Agent to make the automatic selection of genetic operators in the

algorithm is the parameter configuration of the Agent; a good parameter configuration can give us quality solutions, but on the other hand, we do use a wrong parameter configuration for the Agent we will get low-quality solutions.

In this work, the parameter values of the SARSA Lambda Agent have been obtained by assigning values arbitrarily by performing multiple experiments to determine the current values. The source code can be downloaded from[1].

5 Conclusions and Future Work

This work proposes a new adaptive operator selection strategy using a reinforcement learning agent. This SARSA Lambda reinforcement learning strategy has been integrated into a dynamic multi-objective decomposition-based algorithm called DMOEA/D-SL. Furthermore, a state-of-the-art adaptive operator selection strategy, in this case, "Adaptive operator selection with test-and-apply structure for decomposition-based multi-objective optimization (TAOS)," has also been integrated into a dynamic multi-objective decomposition-based algorithm.

In the case of the proposed algorithm, the Agent learns to select the best genetic operator at a given moment, even when the definition of the problem changes over time. Extensive experimentation has been performed, and the results have been evaluated with three metrics: hypervolume, generalized spread, and inverted generation distance.

The Wilcoxon test has been applied with a significance level of 5%. The Wilcoxon test suggests that experimentation results are favorable in two metrics for DMOEA/D-SL. These results suggest that the algorithm produces high-quality solutions with a good approximation to the Pareto front and good dispersion.

Future work proposes exploring the parameters of the SARSA Lambda Agent more broadly to achieve better results in the three metrics used. On the other hand, using more than four genetic operators would test the behavior of the strategies used in this work and their performance.

Finally, other reinforcement learning strategies can also be considered to improve the quality of the solutions generated by the algorithm further.

[1] github.com/JAlfredoBrambila/DMOEAD_SL

Acknowledgments

The authors thank CONAHCyT, Mexico, for the support provided through scholarships for postgraduate studies and the National System of Researchers.

References

1. **Azzouz, R., Bechikh, S., Said, L. B. (2016).** Dynamic multi-objective optimization using evolutionary algorithms: A survey. Adaptation, Learning, and Optimization, pp. 31–70. DOI: 10.1007/978-3-319-42978-6_2.
2. **Li, K., Fialho, A., Kwong, S., Zhang, Q. (2014).** Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation, Vol. 18, No. 1, pp. 114–130. DOI: 10.1109/tevc.2013.2239648.
3. **Pei, J., Mei, Y., Liu, J., Yao, X. (2022).** An investigation of adaptive operator selection in solving complex vehicle routing problem. PRICAI 2022: Trends in Artificial Intelligence, pp. 562–573. DOI: 10.1007/978-3-031-20862-1_41.
4. **Fialho, Á., Costa, L. D., Schoenauer, M., Sebag, M. (2010).** Analyzing bandit-based adaptive operator selection mechanisms. Annals of Mathematics and Artificial Intelligence, Vol. 60, No. 1-2, pp. 250–64. DOI: 10.1007/s10472-010-9213-y.
5. **Brambila-Hernández, J. A., García-Morales, M. Á., Fraire-Huacuja, H. J., Angel, A. B. D., Villegas-Huerta, E., Carbajal-López, R. (2023).** Experimental evaluation of adaptive operators selection methods for the dynamic multiobjective evolutionary algorithm based on decomposition (DMOEA/D). Studies in

Computational Intelligence, pp. 307–330. DOI: 10.1007/978-3-031-28999-6_20.

6. **Sun, L., Li, K. (2020).** Adaptive operator selection based on dynamic Thompson sampling for MOEA/D. Lecture Notes in Computer Science, pp. 271—284. DOI: 10.10 07/978-3-030-58115-2_19.
7. **Dong, L., Lin, Q., Zhou, Y., Jiang, J. (2022).** Adaptive operator selection with test-and-apply structure for decomposition-based multi-objective optimization. Swarm and Evolutionary Computation, Vol. 68, pp. 101013. DOI: 10.1016/j.swevo.2021.101013.
8. **Lin, W., Lin, Q., Ji, J., Zhu, Z., Coello, C. A. C., Wong, K. (2021).** Decomposition-based multiobjective optimization with bicriteria assisted adaptive operator selection. Swarm and Evolutionary Computation, Vol. 60, pp. 100790. DOI: 10.1016/j.swevo.2020.100790.
9. **Goldberg, D. E. (1990).** Probability matching, the magnitude of reinforcement, and classifier system bidding. Machine Learning, Vol. 5, pp. 407–425. DOI: 10.1023/A:1022681708029.
10. **Thierens, D. (2005).** An adaptive pursuit strategy for allocating operator probabilities. Proceedings of the 7th annual conference on Genetic and evolutionary computation, pp. 1539–1546. DOI: 10.1145/1068009.1068251.
11. **Xue, Y., Zhu, H., Liang, J., Słowik, A. (2021).** Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification. Knowledge-Based Systems, Vol. 227, pp. 107218. DOI: 10.1016/ j.knosys.2021.107218.
12. **Sutton, R. S., Barto, A. G. (1998).** Reinforcement learning, second edition: An Introduction. London, England: MIT Press.
13. **Deb, K., Bhaskara, U., Rao, N., Karthik, S. (2007).** Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling. International conference on evolutionary multi-criterion optimization, pp. 803–817. DOI: 10.1007/978-3-540-70928-2_60.
14. **Zhang, Q., Li, H. (2007).** MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation, Vol. 11, No. 6, pp. 712–731. DOI: 10.1109/tevc.2007.892759.
15. **Goh, C., Tan, K. C. (2009).** A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. IEEE Transactions on Evolutionary Computation, Vol. 13, No. 1, pp. 103–127. DOI: 10.1109/ tevc.2008.920671.
16. **Farina, M., Deb, K., Amato, P. (2004).** Dynamic multiobjective optimization problems: test cases, approximations, and applications. IEEE Transactions on Evolutionary Computation, Vol. 8, No. 5, pp. 425–442. DOI: 10.1109/tevc.2004.831456.
17. **Jiang, S., Yang, S., Yao, X., Tan, K. C., Kaiser, M., Krasnogor, N. (2018).** Benchmark problems for CEC2018 competition on dynamic multiobjective optimization. Newcastle University.

Article received on 18/01/2024; accepted on 08/05/2024.
**Corresponding author is Héctor Joaquín Fraire-Huacuja.*

ISSN 2007-9737

Prediction of Enterprise Financial Health Using Machine Learning and Financial Reasons for Taiwan Economic Companies

Daniel Ruelas[1], Fernando-Gaxiola[1,*], Alain Manzo-Martinez[1], Raymundo Cornejo-Garcia[1], Jesus Soto-Vega[2]

[1] Universidad Autónoma de Chihuahua, Facultad de Ingeniería, Mexico

[2] Universidad Autónoma de Baja California, Facultad de Ingeniería, Mexico

dan.rm91@gmail.com, {lgaxiola, amanzo, rcornejo} @uach.mx, jesussoto@uaim.edu.mx

Abstract. In actuality, the financial investment in Enterprises of the world is common. This investment is performed using internet platforms and value markets. This can generate a loss for many investors due to the uncertainty of the future financial health of the Enterprise. A comparison for the prediction of financial health based on algorithms of machine learning, particularly Support Vector Machine (SVM), Artificial Neural Networks (ANN), K-Nearest Neighbors (KNN), and Adaptive Neuro-Fuzzy Networks (ANFIS) is presented. The database of the Taiwan Economic Journal from 1999 to 2009 is used, with 95 financial ratios of enterprises financially healthiest and with bankruptcy problems. In the ANN, the epoch number, numbers of neurons, activation functions in each layer, loss function, and learning rate are tested; also, an architecture of a Convolutional Neural Network (CNN) is implemented. In SVM, the experiments are performed using different kernels, polynomial, RBF, linear. Besides, the size of C, size of gamma, and size of the polynomial are varied. In KNN, experiments with different numbers of neighbors, types of weight, and values of P are realized. In ANFIS, experiments with variants of the numbers of fuzzy rules, quantity and type of membership functions, number of epochs, and input dimensions are performed. Optimization using Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) of the three models with the best results are performed; the optimization is based on the search for the best hyperparameters that would provide a higher accuracy. The neural network models presented the best average for all the proposed tests.

Keywords. Prediction, machine learning, financial.

1 Introduction

Currently, the detection of financial problems through the use of intelligent algorithms is a subject that continues to be investigated. It has been proven that some algorithms can make predictions earlier and more effectively than a professional in the area. For example, Suryawanshi et al. [31], performed the prediction of cryptocurrencies price for Bitcoin, Ethereum, and Ripple using a Long Short-Term Memory architecture (LSTM).

Although the volatility of cryptocurrencies is high and almost unpredictable, this algorithm can help invest. There is also the case of trading, which talks about how Machine Learning (ML) algorithms can generate predictions that are impossible for humans to generate.

This is because trading operations are too fast for a human. In addition to the fact that, combined with a large amount of data, it reduces risks and obtains greater benefits.

It is estimated that, currently, 4 out of 5 trading operations are done automatically. For this task, Chen et al. [5] generated a trading algorithm using the Light Gradient Boosting Machine (LightGBM) algorithm to construct the minimum variance portfolio of the mean-variance model with a Conditional Value at Risk (CVaR) constraint, to generate an efficient investment portfolio.

Finding an effective solution for the issue of predicting financial bankruptcies is important for

different areas, be it business, government, or even social. The main reason for this work is to detect financial difficulties early, using machine learning algorithms, to avoid financial investments in companies that are at imminent risk of bankruptcy.

First, the concept of bankruptcy or financial bankruptcy in a company must be clarified. In López [21], the concept of bankruptcy is defined as the company's inability to meet its debts with the available resources, so it must cease its activities immediately. In little words, net patrimony is negative, when the total value of assets is not enough to pay off creditors. The main characteristics of bankruptcy according to López [21] are:

1 Situation of irreversible or permanent disappearance: It occurs when the company declares bankruptcy and is in the process of disappearing. It is permanent.

2 The assets are less than the liabilities: It occurs when in a company, the debts exceed the assets.

3 Affects the entire company: Creditors dispute parts of the company and legally affect the company as a whole. General bankruptcy can be avoided by selling subsidiaries.

4 It must be legally classified: It is a situation provided by law, to avoid any fraud.

In this work, a financial analysis is proposed, which is given from the observation of patterns of the financial ratios of companies. These financial reasons come from the financial models or basic financial statements.

The set of basic financial statements, the income statement, cash flow, and balance sheet are the general x-rays of the company and these are the financial ratios to use.

Different financial ratios use these basic financial statements to define the financial health of a company. Guajardo and Andrade [13] present some financial ratios, such as the acid test and accounts receivable turnover. The first ratios show how much liquidity the company has, so it divides the product of the subtraction of the current assets account (balance sheet) and inventories (balance sheet), between the current liabilities (also from the balance sheet).

Account turnover indicates how many times a year the accounts provided by the company are rotated. It is obtained by dividing net sales (income statement) by accounts receivable (balance sheet). As well as these examples, 95 financial ratios are raised in the investigation, which will be discussed later.

Santoso and Wibowo in 2018 [27], use machine learning models such as K-nearest neighbors, neural networks, support vector machines, and neuro-fuzzy networks, to classify a company as stable or bankrupt. Here are some examples of related work using similar algorithms and architectures:

Xie, Lu, and Yu in 2011 [39], mentioned in their research that they can predict the bankruptcy of a company using SVM, with variables similar to the previous ones, such as financial profitability and return on investment.

In 2017, Mselmi et al. [22], used logistic regression, neural networks, SVM, least squares, and a hybrid model of least squares with SVM to classify companies with possible bankruptcy.The main contributions of this research are as follow:

– Identify a functional financial dataset with financial ratios adequate for the research.
– Implementing and testing of Adaptive Neural Fuzzy Inference System (ANFIS), Neural Networks (NN), K-Nearest Neighbor (KNN), and Support Vector Machine (SVM) to classify the financial state of a company through the use of financial ratios, derived from the basic financial statements.
– Optimization of the models implemented using bio-inspired optimization algorithms, like Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), to improve the results for the classification.

Currently, technology is generating better living conditions and knowledge that helps in decision-making. Within this technology, machine learning algorithms are integrated in different areas. This type of technology has been used in most areas of daily life due to more efficient and accurate algorithms, and more capable hardware [25].

This research is important in the business, government, and social spheres since it has been proven that classification can be made using machine learning algorithms, with a degree of confidence greater than the financial experts in the area. With this, strong financial decisions can be made that solve problems such as knowing if it's possible to invest in a company and if it is redeemable through credit, among other things.

Currently, the industry does not have a sufficient supply of solutions on the market, therefore it is still being investigated, so the objective of this and other investigations is to risk people's assets as little as possible and generate knowledge that can be used for future research.

Research related to the investigation topic of the paper is described as follows:

Stasko et al. [29], utilized the Altman financial model to calculate a future prediction of bankruptcy in Companies from Letonia using 5 financial ratios; the result of this method is a probability of imminent bankruptcy, gray zone, and secure zone.

Tabbakh et al. [32], used a dataset from Polonia with 43,405 companies; For this case, remove the instances that have null data, normalization, and the "SMOTE" technique are used for preprocessing, besides the SVM model for prediction gave a 98.8 % in accuracy.

Kansal and Sharma [18], mentioned the use of SVM and Neural network models for the prediction in small and medium-sized companies of a database from France.

Arieshanti et al. [2], used a database of 240 companies with 30 financial ratios, and for the prediction implemented an SVM model with a lineal kernel, variable C of 1, and a neural network with sigmoidal activation function and 5,000 epochs for training.

Xie et al [39], used 260 Chinese companies with 28 financial ratios, half of the companies are bankrupt and half are not.

Two SVM models are implemented, first with the aforementioned database and another with aggregate corporate governance and external market variables.

Narvekar and Guha [24], proposed a SVM model for prediction in the database "Compustat". This database contains 75 financial ratios from 21,114 American companies, of which there are 1,212 companies in bankruptcy and 19,902 stables.

For preprocessing, null values are eliminated which allows removal of 18 financial ratios, and the SMOTE technique to balance the data. Santoso and Wibowo [27], proposed an SVM model with a linear kernel to perform a prediction in a database of Indonesian companies with 20 financial ratios.

Shetty et al. [28], proposed SVM and Neural network models for the prediction in small and medium-sized companies of a database from Belgium. Wang [37], proposed an Artificial Neural Network model for prediction in a database called "Qualitative Bankruptcy", which uses disconnection of neurons, truncation technique, softmax activation function, Adam optimizer, and categorical cross entropy loss function.

Abdou et al. [1], used a neural network model to be implemented in a database of 14 financial and 3 non-financial indicators of companies registered with banks in the Midwest.

Sudarsanam [30], proposed a neuro-fuzzy network model that used only the variables of the Altman method; this model is implemented in a database of 125 companies from the Indian economic monitoring center, where some of them are bankrupt.

Arora and Saini [3] proposed an ANFIS model with Altman's variables and 3 bell-type membership functions; for the prediction, 1,000 companies and a total of 4 years of maximum prediction are used.

Muslim and Dasril [23], used a KNN model on a database of companies in Poland, with 65 financial ratios, and the data are normalized and scaled.

In this work, take in consideration the literature, the implementation of several models of machine learning and a selected number of financial ratios is performed to achieve the prediction of bankruptcy.

The rest of this paper is organized as follows: Theoretical background is presented in Section 2, in which the theory of the financial statements and computational models are presented; the methodology implemented is given in Section 3.

Experiment analysis is presented in Section 4 and, Section 5 and 6 gives the Discussion and final Conclusion.

Table 1. Examples of financial ratios

Financial Statement	Ratio
LIQUIDITY	Current ratio
	Acid test
ACTIVITY	Inventory Turnover
	Average Inventory
	Accounts receivable turnover
	Average collection period
	Turnover of current assets
	Fixed asset turnover
	Total asset turnover
	Debt ratio
	Interest coverage
INDEBTEDNESS	Gross profit margin
	Operating sales margin
	Net profit margin
COST EFFECTIVENESS	Return on operating investment
	Return on total investment
	Income on capital

2 Theoretical Background

2.1 Financial Theory

In the financial field, it must be clarified which are the basic financial statements and the accounts that are within the financial statements. The following are defined as basic financial statements: balance sheet, income statement, statement of changes in equity, changes in financial situation, and cash flow [7].

In this work, only the balance sheet, the income statement, and the cash flow are worked on. This is because the financial ratios are calculated exclusively with the accounts that are within these financial statements.

2.1.1. Basic Financial Statements

The three basic financial statements are described as follows:

- "The balance sheet is the accounting document that reports on a specific date on the financial situation of a company, where the obligations, capital, properties in monetary value, and rights are presented".
- "The income statement or profit and loss statement is defined as the document that provides detailed information on where the profit or loss of the accounting year is obtained."
- "Cash flow is the statement that shows the movement of income, expenses and the availability of funds on a given date".

2.1.2. Financial Ratios

The financial ratios come directly from the three basic financial statements, which represent a general overview of the company's situation, which means that you can consult the information of the entire company (financially speaking) in this executive summary (the three basic financial statements).

At the same time, these statements are those used by financial experts to generate some type of evaluation, either directly or with some financial elements involved, derived from cash flow, income statement, and balance sheet. In Table 1, Examples of financial ratios are shown [7].

The financial ratios are important to define a bankruptcy; because these can be compared in percentages or small values, which do not vary concerning the size of the company but vary in its real financial performance, in the defined time.

Stasko et al. [29], mention that there is a formula or indicator to predict bankruptcy, called Altman's Z-score. According to Vera [34], this model provides an accuracy of between 80% and 90% to know if a company is bankrupt or not.

2.2 Computational Models

As for the computational part, Neural Networks (ANN) [8, 16], Convolutional Neural Networks (CNN) [12, 41], Adaptive Neuro-Fuzzy Inference Systems (ANFIS) [16,17, 19], Artificial, Support Vector Machines (SVM) [36, 38, 26], and K-Nearest Neighbors (KNN) [6, 33] are implemented for the Prediction of Enterprise Financial Health. For optimization, Genetic Algorithms (GA) [40, 14]

and Particle Swarm Optimization (PSO) [40, 9, 15] are used.

3 Methodology

The methodology used is focused on finding the best combination of parameters that made the machine learning algorithms work optimally, implemented on the same database.

3.1 Database

The database of Taiwan is used to test all the models, which contain 6,819 companies, labeled with 1 the 6,599 financially stable companies, and with 0 the 220 bankrupt companies, with 95 financial ratios (described in Table 16) [20]. These 95 financial ratios are used as attributes for inputs to train and test the machine learning models.

The data on these companies are from the Taiwan economic journal, from 1999 to 2009. This database is considered highly unbalanced because the bankrupt companies barely represent 3% of the total, which requires a dimension reduction or augmentation to balance the database [35].

3.2 Data Pre-Processing

Scaling and standardization of the data are performed to improve the performance of the algorithm using the Standard Scaler, which means putting the data with a mean of 0 and variance of 1. The variables are on different scales and by applying the standard scaler technique they remain on a similar scale.

3.2.1. Database Balance

A data balance is generated in the database so that it had the same number of bankrupt and stable companies. This means that an algorithm is used to increase the size of the database and thereby generate more information.

In this way, there are 6,599 bankrupt companies and 6,599 stable companies. For this purpose, the SMOTE technique is used [4]. This is used to increase the performance of the algorithms so they can identify patterns that are necessary to classify companies.

Table 2. Parameters of the chromosome for GA

Algorithm	Parameters
Support Vector Machine (SVM)	Kernel: RBF, Polynomial
	C: 2^-5 – 2^15
	Gamma: 2^-15 – 2^3
	Polynomial dimension: 3 - 6
	Chromosome size: 4
	Crossing probability: 70%
	Mutation probability: 25%
K-Nearest Neighbour (KNN)	Weights: Uniforms and distance.
	Number of neighbors: 1-10
	P: 1-2
	Chromosome size: 3
	Crossing probability: 70%
	Mutation probability: 33%
Artificial Neural Network (ANN)	Activation function: Relu, Linear, Sigmoidal, Tanh, Prelu, Selu, Elu.
	Learning rate: 0.01, 0.001, 0.0001, 0.00001.
	Optimizer: Adam, RMS, SGD, Adadelta, Adagrad, Adamax, Nadam, Ftrl.
	Number of layers: 1, 2, 3
	Number of neurons per layer: 1-99
	Chromosome size: 10
	Crossing probability: 70%
	Mutation probability: 10%
	Activation function: Relu, Linear, Sigmoidal, Tanh, Prelu, Selu, Elu.

Specifically, in this case, there are few bankrupt companies, so it is more difficult for the model to detect patterns of bankrupt companies in the training phase. Therefore, increasing the number of samples helps to train better.

3.2.2. Dimension Reduction

Within the parameters to manipulate, dimension reduction is used. This is often used to generate better performances for classification algorithms. In this particular case, the PCA algorithm is used. This algorithm is focused on the variance of the components of the database. Values are assigned according to the level of variance and the number of values to be searched is chosen.

Each algorithm is tested with different dimensions. Particularly for Artificial Neural Networks, KNN, and SVM, the dimension is reduced to 93, which is the dimension that generated the best results in the experimentation. It is a reduction of 2 dimensions, which caused some noise in the data. In ANFIS, the best result is provided by PCA with a value of 30.

3.2.3. Data Separation

The data from the database are separated into training and testing sets. The separation is 80% for training and 20% for testing. This number is used to avoid underfitting.

3.3 Classification Algorithms

The classification algorithms are: Artificial Neural Networks, KNN, SVM, and ANFIS. The metric to evaluate them is accuracy. Each of the methodologies of these algorithms is detailed below.

3.3.1. Support Vector Machine (SVM)

The parameters used in the SVM model are the RBF kernel. This kernel requires two parameters, C and Gamma. For this case, C=1000 and gamma =0.01 are used. These parameters are determined, after experimenting with different values of C and Gamma; this is the best architecture found.

3.3.2. K-Nearest Neighbors (KNN)

The parameters used in the KNN model are 1 in the number of neighbor and, “uniform” for the weights.

3.3.3. Artificial Neural Networks (ANN)

In the Artificial Neural Networks models, two options are used: convolutional and multilayer Neural Networks. The parameters used in the convolutional networks are: A 1-dimensional convolution layer, with 32 filters of size 8 and a relu-type activation function.

Another Max pooling layer with a pool size of 2 and stride none. A layer of flattening is also added. After this, a fully connected layer is defined, with 93 neurons and a RELU activation function.

Finally, an output layer with 2 neurons and RELU activation. We added 93 neurons in the fully connected layer since that is the number of ratios used. The loss function used is binary cross-entropy, since there are only two output classes, and the optimizer used is “Adam”. The model is trained with 100 epochs and a batch size of 16.

As for the multilayer neural network, the architecture is as follows: an input layer of 100 neurons, and a hidden layer of 100 neurons, both with RELU activation. Finally, a 1-neuron output layer with sigmoidal activation. The loss function used is binary cross-entropy, given that there are only 2 possible outputs and the Adam optimizer. It is trained with 100 epochs, and a batch of 10 is used.

3.2.4. Adaptive Neural Fuzzy Inference System (ANFIS)

The following parameters are used in the ANFIS model. In GENFIS, the partition is called a grid, which generates the greatest number of possible combinations of membership functions. Gaussian membership functions are used, which are best adapted to the type of data used. 40 epochs are used for training.

It is configured with the hybrid mean square error and backpropagation model. In GENFIS, a change is made to the generation of rules, to generate less and make the model faster and more efficient.

The rules are set equal to the number of membership functions, rather than the dot product of them.

3.4 Cross Validation and Confusion Matrix

A cross-validation of 10 is applied to generate better confidence in the algorithms, and prevent them from being the results of the arrangement of some data. Confusion matrix is used to validate the

Table 3. Results of MNN, CNN, KNN, SVM and ANFIS in accuracy values

Algorithms	Accuracy
Multi-Layer Neural Network (MNN)	98.86%
Convolutional Neural Network (CNN)	99.28%
K-Nearest Neighbors (KNN)	97%
Support Vector Machine (SVM)	98.40%
Adaptive Neural Fuzzy Inference System (ANFIS)	81.82%

Table 4. Confusion matrix of MNN, CNN, KNN, SVM and ANFIS (TP: True Positives, TN: True Negatives, FP: False Positives, FN: False Negatives)

Algorithms	TP	TN	FP	FN
MNN	1,304	1,306	30	0
CNN	1,315	1,306	19	0
KNN	1,245	1,304	89	2
SVM	1,294	1,304	40	2
ANFIS	1582	1658	415	297

efficiency of the algorithms in bankrupt and stable companies' classifications.

3.5 Optimization of Classification Algorithms

Genetic Algorithms and PSO are used to optimize the classification algorithms. Only the optimization is applied to Artificial Neural Networks (ANN), SVM, and KNN, since they are the models that generated the best predictions by more than 12% difference, concerning ANFIS and convolutional networks [10, 11].

3.5.1. Genetic Algorithms (GA)

The parameters used for optimizing the Multilayer Neural Networks architecture and KNN with genetic algorithm are: 100 generations, 200 individuals, one-point crossover, tournament selection method with size 6, real-type chromosome, and the fitness function to maximize the accuracy. In the case of SVM, it is limited to 30 generations, due to the computational capacity required by the algorithm. Table 2 shows the parameter ranges for each real-type gene that makes up each individual or chromosome.

3.5.2. Particle Swarm Optimization (PSO)

The hyper-parameters and parameters of the chromosomes used for optimizing the KNN, SVM, and ANN are the same as the ones implemented in GA.

4 Simulation Results

The results are obtained in two stages. The first stage is achieved through experimentation and knowledge of the literature. In the second stage, the optimization of the models is performed. In both cases, accuracy, F1-score, confusion matrix, and precision metrics are used to verify the effectiveness of each model.

The results are validated using K-fold, to avoid having a performance resulting from the memorization of the algorithms or randomness of the data. Finally, a statistical validation is performed to determine if the results are significantly different. Additionally, the optimizations, the architectures and accuracy obtained in each experiment are shown.

4.1 Results with Experimentation

Table 3 shows the results obtained in terms of accuracy of the four algorithms to be compared.

It is observed that the best result is obtained by the convolutional networks algorithm (CNN). For just under half a percentage point, the multi-layer Neural Networks (MNN) is in second place.

In general, all algorithms exceeded 97%, except the ANFIS algorithm, which is positioned almost 16 points below the KNN algorithm. It can be determined that there is a high efficiency in all algorithms.

Table 4 presents results obtained from the confusion matrix; it can be seen that except for ANFIS, all the algorithms have few false negatives. This means that algorithms rarely make a mistake when classifying a company as "bankrupt".

Most of the errors are found in false positives, which means that there are some companies classified as stable, which are bankrupt.

Table 5. Precision of MNN, CNN, KNN, SVM and ANFIS

Algorithms	Accuracy
MNN	99%
CNN	99%
KNN	97%
SVM	98%
ANFIS	82%

Table 6. F1 score of MNN, CNN, KNN, SVM and ANFIS

Algorithms	Accuracy
MNN	99%
CNN	99%
KNN	97%
SVM	98%
ANFIS	82%

Table 7. K-fold of 10 for MNN, CNN, KNN, SVM and ANFIS

K-Fold	MNN	CNN	KNN	SVM	ANFIS
1	98.94	50	93.86	93.25	55.83
2	99.17	53.26	96.59	94.86	50.45
3	98.79	88.56	95	93.10	38.18
4	99.09	50	93.56	91.49	43.33
5	99.55	84.09	97.57	95.16	48.79
6	99.47	71.59	98.33	96.48	52.58
7	99.09	56.36	97.04	96.48	40.68
8	99.39	67.27	97.50	95.60	39.02
9	98.86	50.04	98.18	96.62	39.31
10	99.47	60.65	97.19	95.59	37.45

In Table 4, the results of the Confusion Matrix are consistent with the accuracy and in the case of CNN and ANN, they only show failures in false positives, this means bankrupt companies are detected but they are not in bankruptcy.

Normally in databases for predicting financial bankruptcies (for example, those of related work), there are many false positives and false negatives, since the normal thing in these databases is that they are highly unbalanced, so the result of 30 and 19 false positives for CNN and ANN respectively, can be interpreted as a good result. We should also not forget to mention that the false negatives in the case of CNN and ANN are 0.

Regarding KNN, it can be seen that there are 89 false positives and 2 false negatives. Regarding SVM, 40 and 2 are obtained respectively. This means that both are quite competent for the type of data that is handled. The ANFIS algorithm, on the other hand, has many false positives (415) and false negatives (297), so it is another verification that it is not the most suitable for this case.

Tables 6 and 7 show the values obtained for precision and f1 score. The results are consistent and practically the same as those previously obtained with accuracy. This is because precision is calculated by dividing true positives by multiplying true positives and false positives. It can be seen that there are very few false positives in all algorithms except ANFIS (approximately 2 to 6%). In the case of F1 score, it is also based on false positives and false negatives, which are very low, so the results do not vary significantly.

The K-fold cross-validation technique is used to validate the algorithms, it is necessary to obtain accurate results and not products of random situations.

Table 7 shows the results obtained by K-fold of 10. In general, it can be observed that, except in the case of MNN, all the algorithms decreased the accuracy score concerning the results in Table 3. However, in the case of CNN and ANFIS, extremely fluctuating and low results are shown, compared to Table 3.

Given these results, it can be seen that neither of the two algorithms managed to have a result similar to that of the test without k-fold. As for MNN, there are results even greater than those obtained in the experimentation phase without k-fold.

For SVM and KNN, there are results very close to those obtained in the previous phase. Table 8 shows the average k-fold of 10. This can be interpreted as the true efficiency of the algorithms since they are tested at different data arrangements, where enough experiments are generated to determine the real effectiveness of the algorithms. It can be seen that MNN increased its effectiveness from 98.86% to 99.18%.

Table 8. Average of K-fold of 10 for MNN, CNN, KNN, SVM and ANFIS

MNN	CNN	KNN	SVM	ANFIS
99.18	63.18	96.48	94.86	44.56

Table 9. Results of Friedman test

Sums	ANN	SVM	KNN
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
-	3	1	2
Sum	**30**	**10**	**20**
Sum Square	**900**	**100**	**400**

Table 10. Wilcoxon rank results

Average	ANN	SVM	KNN
-	23	3	5
-	26	6	13
-	21	2	7
-	24.5	1	4
-	30	8	18
-	28.5	11.5	20
-	24.5	11.5	15
-	27	10	17
-	22	14	19
-	28.5	9	16
Average	**25.5**	**7.6**	**13.4**

This means that it is an efficient algorithm and it has been the one that has given the best results in terms of accuracy and adapted best to changes, so, it is not a product of memorization or over-training. In the case of SVM, a reduction of almost 4 points can be seen, where it continues to remain above 90%. However, after the K-Fold, at this stage, it can be determined that its ranking has been lower than KNN, which lost half a percentage point.

Finally, it should be mentioned that ANFIS presented a very low result, where it does not even reach 50% after the k-fold, therefore, it can be seen that it only worked with one data arrangement and that it was probably memorizing the data that are given to him, so when it was subjected to this technique, his performance dropped greatly.

In the case of CNN, it became the second worst algorithm and the explanation is that by moving the data, CNN was not able to learn efficiently. Probably due to the filtering and pooling techniques, they generated a complicated model to work with, without forgetting that it is an algorithm generally designed to work with image bits with binary numbers and not with the type of information that financial ratios have.

4.2 Statistical Verification of the Results with Experimentation

The Friedman test with the Chi-square statistic is used to statistically verify which algorithm is the best. For this test, a minimum of 3 variables are required. The test is performed with the 3 best algorithms and their respective K-Fold analyses of 10 entries, ANN, KNN and SVM.

This is because the difference between the other two algorithms is too wide and would only introduce noise into this test. The null hypothesis is that the mean of each population is the same and the alternative hypothesis is that at least one is different. The specific result is shown in Table 9:

Therefore, N is equivalent to 10, K equals 3, Q equals 20, and the value of P is 0.000045. Since the P value is less than 0.05, the null hypothesis is rejected.

It can be determined that they are significantly different and that the neural network is significantly better. A Nemenyi test is also performed. The values in the Wilcoxon range are shown in

Table 11. Results of the optimization with GA for ANN, SVM and KNN in accuracy values

Experiment	ANN	SVM	KNN
1	99.50	99.16	97.46
2	99.43	98.90	97.42
3	99.43	99.24	97.42
4	99.39	99.20	97.46
5	99.39	99.20	97.46
6	99.18	99.31	97.42
7	99.30	99.24	97.46
8	99.43	99.20	97.50
9	99.39	99.24	97.42
10	99.43	99.16	97.38
Average	**99.39%**	**99.19%**	**97.44%**

Table 12. Results of the optimization with PSO for ANN, SVM and KNN in accuracy values

Experiment	ANN	SVM	KNN
1	99.50	99.01	97.15
2	99.39	98.90	97.12
3	99.35	98.82	97.08
4	99.70	98.82	97.19
5	99.50	98.90	97.19
6	99.35	98.86	97.19
7	99.50	99.01	97.12
8	99.46	98.75	97.19
9	99.58	98.82	97.12
10	99.39	98.82	97.12
Average	**99.47%**	**98.87%**	**97.15%**

Table 10. The critical value of the Nemenyi test for alpha of 0.05, infinite N, and K of 3 is 9.22612. When calculating the difference in means of the Wilcoxon values, the following is determined:

- The mean difference between ANN-SVM is 17.90 > 9.22612, therefore there is a significant difference.
- The mean difference between ANN-KNN is 12.10 > 9.22612, therefore there is a significant difference.
- The mean difference between KNN-SVM is 5.80 < 9.22612, therefore there is no significant difference.

4.3 Results for Optimization with GA and PSO

ANN, SVM, and KNN are optimized with GA and PSO. The results are reported below. The Table 11 shows the results obtained through GA. In Table 11, it can be seen that SVM and ANN maintained results greater than 99%. There is, therefore, a successful search for parameters, since in all cases the results were improved to those obtained in the experimentation phase shown in Table 3.

It can also be observed that there was an increase in KNN, that is, it remained in a very similar range all the time. It must be remembered that there are very few parameters in this algorithm, so it is consistent that they remain in similar ranges. The Table 12 shows the results obtained through PSO. The best architecture of each of the algorithms was the following:

- ANN: accuracy of 99.50%, first hidden layer with 94 neurons and activation function relu, first hidden layer with 1 neuron and activation function RELU, Adam optimizer, learning rate of 0.01, and output layer with 1 neuron and function sigmoidal activation.
- SVM: accuracy 99.31%, C=64, gamma of 0.125, and RBF kernel.
- KNN: accuracy: 97.50%, 2 neighbors, uniform weight and P=1.

In Table 12, it can be seen that ANN maintained results greater than 99%. There are also 2 results greater than 99% in SVM. In the case of KNN, there are very small improvements, concerning the experimentation phase, but these results are higher. Therefore, there is a successful search for parameters, since in all cases it was possible to improve the results to those obtained in the experimentation phase shown in Table 3. The best

Table 13. K-fold of 10 for ANN-PSO, SVM-GA, and KNN-GA

K-Fold	ANN-PSO	SVM-GA	KNN-GA
1	98.56	97.50	94.92
2	98.56	98.56	97.34
3	98.81	98.63	95.53
4	98.71	97.04	97.72
5	98.85	98.86	98.40
6	98.99	99.69	93.56
7	98.92	99.31	98.03
8	98.96	99.16	97.72
9	99.01	99.54	98.48
10	98.95	99.46	98.18
Average	**98.83%**	**98.78%**	**96.99%**

Table 14. Results of Friedman test for optimized algorithms

Sums	ANN	SVM	KNN
Sum	23.5	25.5	11
Sum Square	552.25	650.25	121

Table 15. Wilcoxon rank results for optimized algorithms

Average	ANN	SVM	KNN
Average	19.70	20	6.8

architecture of each of the algorithms is the following:

- ANN: accuracy of 99.70%, a first hidden layer with 100 neurons and RELU activation function, a second hidden layer with 100 neurons and Tanh activation function, a third hidden layer with 1 neuron and RELU activation function, Adam optimizer, learning rate of 0.001 and output layer with 1 neuron and sigmoidal activation function.
- SVM: accuracy 99.01%, C=4096, gamma of 0.0125, and RBF kernel.
- KNN: accuracy 97.19%, 2 neighbors, uniform weight and p=2.

The 10-fold cross-validation is used for the best-optimized algorithms of ANN, SVM, and KNN, these being ANN-PSO, SVM-GA, and KNN-GA. Table 13, shows the results of each fold.

The ANN is placed within a range of difference in the results of half a percentage point; While the KNN has up to 5 points of difference; Regarding the SVM classifier, a maximum of 99.46 and a minimum of 97.04 are also observed, almost 2 and a half points difference.

Improvements are obtained in KNN and SVM concerning the average obtained without optimization, so both algorithms achieved the objective. In the case of ANN, the result is lower, therefore, it cannot be determined that a better architecture was obtained.

4.4 Statistical Verification of the Results for Optimization with GA and PSO

The Friedman and Nemenyi test are used to statistically verify which algorithm is the best, with the same values and parameters used in the subsection 5.2. The specific result for the Friedman test is shown in Table 14:

Therefore, N is equivalent to 10, K equals 3, Q equals 12.35, and the value of P is 0.002080. Since the P value is less than 0.05, the null hypothesis is rejected. It can be determined that they are significantly different and that the neural network is significantly better.

For the Nemenyi test, the values in the Wilcoxon range are shown in Table 15.

The critical value of the Nemenyi test for alpha of 0.05, infinite N, and K of 3 is 9.22612. When calculating the difference in means of the Wilcoxon values, the following is determined:

- The mean difference between ANN-KNN is 12.90 > 9.22612, therefore there is a significant difference.
- The mean difference between ANN-SVM is 0.30 > 9.22612, therefore there is no significant difference.
- The mean difference between KNN-SVM is 13.90 < 9.22612, therefore there is a significant difference.

5 Discussion

In terms of accuracy (Table 3), the best algorithm is CNN, with 99.28%, followed by ANN with 98.86%. It can also be seen that SVM and KNN had outstanding results since they obtained an accuracy of 98.28 and 97%, respectively. In general, both models can be used due to their accuracy above 97%, which makes them reliable.

On the other hand, ANFIS is placed at 81.82%, so for the moment it can be ruled out as a good algorithm for this database, given the tested architectures.

These data do not yet demonstrate statistical differences; a K-fold of 10 was required to generate statistical tests.

Regarding the precision parameter (Table 5) and the F1 score (Table 6), they are added to analyze if there is any important difference; however, the results are the same as discussed above, there are no variations in comparison with accuracy. This is because both measures are calculated through false positives and false negatives.

K-Fold cross-validation is used to validate the previous data, with size 10, as shown in Table 7. In this case, there are important variations.

First of all, the CNNs in this test did not obtain results higher than 89%. Likewise, it can be seen that the results are very fluctuating and are between 50% and 88.56%. This generated an average of 63.18%.

This variation can be explained because the weights in the CNNs are random; this means that perhaps, given some weights and given the training and testing data structure, the CNN can memorize the data.

Another algorithm that behaved extremely inefficiently is ANFIS, which when applying the K-fold, its performance dropped to 44.56% on average. It may be due to memorization as in the previous case.

Also, some membership functions could have been better adapted to the data of the first test, but in the consecutive tests, the algorithm is no longer as efficient.

In SVM there is also a significant reduction, of just over 3 percentage points. In this case, it is placed at an average of 94.86%, so it is possible to determine that it is still a good algorithm, with a good architecture for predicting financial bankruptcies, given this database.

In KNN there is also a slight reduction, of half a percentage point. In this case, the reduction is minimal, so it can be determined that given the architecture used and given this database, it is an extremely efficient algorithm to predict a financial bankruptcy.

If a final product is developed that requires little computational capacity, this algorithm would be the best option, since it consumes very few computational resources and generates an efficient result.

Regarding the ANN, there are improves, as seen in Table 8, on average it is achieved at 99.18%; This means, in the k-fold test the ANN showed the best average of prediction of financial bankruptcy.

The Friedman test is used for the three best algorithms to generate a statistical verification, with the results obtained by the K-fold validation.

It can be seen in the previous section that a result of P of 0.000045 is generated, which means that the null hypothesis is rejected since it is less than 0.05. This hypothesis says that the population average is the same in the three algorithms, therefore, at least one is different.

Regarding the Nemenyi test, it can be determined that significant differences occur between ANN, concerning SVM and KNN with a 5% significance, given that it is less than 0.05.

In the optimizations phase, ANN, SVM, and KNN are used, since the difference between CNN and ANFIS is great and the execution time would be much higher. Regarding ANNs, the best result obtained in GA is 99.50% in terms of accuracy.

Likewise, in SVM 99.20% is obtained and in KNN 97.46%. Comparatively to the results of the first test, an increase of almost 1 percentage point can be seen in the ANN and in SVM, as well as half a percentage point in KNN.

In PSO, the best value in terms of accuracy is obtained, since 99.70% is achieved in the ANN, 99.01% in SVM, and 97.15% in KNN.

Table 16. Financial ratios from the database of Liang and Tsai [20]

Return on total assets before taxes	Return on total assets after taxes	Return on assets before interest and depreciation, after taxes
Gross operating margin	Gross sales margin	Operating profit rate
Net interest rate before taxes	Net interest rate after taxes	Non-operating net income ratio.
Continuous interest rate	Operating Expense	Rate Research and development expense rate
Cash flow rate	Interest rate on interest-bearing debt	Effective tax rate
Net value per share (A)	Net value per share (B)	Value per share (C)
Earnings per share for the last four seasons	Cash flow per share	Earnings per share
Operating profit per share	Net earnings per share before taxes	Sales gross profit growth rate
Operating profit growth rate	Net profit growth rate after tax	Regular net profit growth rate
Continuous growth rate of net income	Total assets growth rate	Net worth growth rate
Total Asset Return Growth Rate Ratio	Cash Reinvestment Percentage	Current Radius
Acid Test	Interest Expense Ratio	Total Debt/Net Worth
Debt Ratio	Net Worth/Assets	Long-Term Fund Suitability Index
Debt dependence	Contingent liabilities / net worth	Operating profit / paid-in capital
Net income before taxes / paid-in capital	Inventory and accounts receivable	Total asset turnover
Accounts Receivable Turnover	Average Receivable Days	Inventory Turnover Rate
Frequency of fixed asset turnover	Net worth turnover rate	Income per person
Operating profit per person	Allocation rate per person	Working capital to total assets
Quick assets / total assets	Current assets / total assets	Cash / total assets
Quick assets / current liabilities	Cash / current liabilities	Current liabilities with assets
Operating funds to liabilities	Inventory/working capital	Inventory / current liabilities
Current liabilities/liabilities	Working capital/equity	Current liabilities/equity
Long-term liabilities with current assets	Total income / total expenses	Expenses / total assets
Current asset turnover rate	Rapid asset turnover rate	Working capital turnover rate
Cash turnover ratio	Cash flow to sales	Fixed assets to assets
Current liability to liability	Current liability to equity	Equity to long-term liability
Cash flow to total assets	Cash flow to liabilities	Cash flow from operations to assets
Cash flow to equity	Current liabilities with current assets	Liability-asset mark
Net income to total assets	Total assets to price	Total assets to price of gross domestic product
Interval without credit	Gross profit on sales	Net profit from stockholders' equity
Liabilities versus equity	Degree of financial leverage	Interest coverage ratio
Net income indicator	Shareholders' equity to liabilities	

The results in this case are consistent. In the optimizations, the execution times are extremely high. The ANNs took about 1 week per run, of which, 10 are done for PSO and 10 for GA. Likewise, in SVM, the generation reduction had to be done, since some polynomial kernel architectures could not complete the execution, due to the level of complexity and the penalty that is given.

The KNN algorithm is the fastest in this aspect since it could have executions of 6 hours or less, depending on the machine where the algorithm is run. Finally, a K-Fold is made of the best-optimized results, where the surprise is that the ANN architecture is not better than the architecture obtained in the empirical experimentation phase.

This, like the result of the CNNs, could be because the model in terms of accuracy managed to obtain a superior result through memorization, which in the end, if these data are moved, it does not generate such efficient results.

Either way, the result is 98.83%, which generates a good and usable architecture. In KNN there is an improvement of 0.3% and in SVM 3.92%. SVM had the greatest increase in optimizations and helped generate a model that is almost on par with the ANN. This means that validations or tests can be done with both algorithms, whenever greater certainty is required.

Regarding statistical validation, the Friedman test is also used with the K-fold of 10, where a P value less than 0.05 is obtained, which rules out the null hypothesis. This means that the differences are statistically significant.

In the Nemenyi test, it can be observed that significant statistical differences occurred between ANN and KNN. There is also a significant difference between KNN and SVM. In both cases, Nemenyi values lower than 0.05 are generated, which is the level of significance. Regarding ANN and SVM, it can be stated that they do not have a significant difference, since their value was higher than the significance level of 0.05.

Given these statistical results, it can be determined that the ANN shows the best results, with a significant statistical difference. Finally, comparing the values obtained, the best result in accuracy terms is obtained by the optimized ANN-PSO, with an accuracy of 99.70%, followed by the optimized SVM-GA value of 99.24%, and finally the optimized KNN-GA value of 97.50%; the best result in accuracy average with k-fold of 10 is obtained by the ANN of empirical analysis with 99.18% value, followed by the SVM-GA value of 98.78%, and the KNN-GA value of 96.99%.

6 Conclusions

Based on the analysis of the results, it can be concluded that the model that has the best performance in predicting financial bankruptcy in the database of Taiwanese companies is the ANN, with 99.18% accuracy values and validating their performance using cross validation K-Fold.

This model has a high computational cost since it is a network with a very high number of connections, so its use would require a little more resource than less precise models. Naturally, it has the advantage of having greater precision, when doing a deep analysis, than other models.

Analyzing in terms of computational cost, execution time, and accuracy of the algorithm, it could be determined that KNN (96.73%) is a good option. KNN is an algorithm that is used on a large scale due to its low computational cost and low complexity, so it would have that advantage on its side.

However, the accuracy shown by ANN would be sacrificed. It should be noted that the computer where the optimization algorithms are executed is not a high-processing one, as it has a 2.90 GHz Intel I7 processor, 16 GB of RAM and a graphic card of 2 GB.

The greatest justification for this work is to generate a functional model that provides greater certainty to the investments of businessmen, governments, and people in general.

With the results provided by these computational models, one can be certain that this objective is covered, in Taiwanese companies that are listed on the stock exchange and that have basic financial statements, with 99.18% effectiveness.

It can also be concluded that the general average of 80% reliability of the “Altman” financial prediction tool was exceeded.

The optimizations generated better results in KNN and SVM with little difference, but no improvement in ANN. The time taken in the

executions must be evaluated with the results obtained, as well as the carbon footprint generated by having a computer on for a long time.

Also, in this area, it should be considered for future research, not to base the optimizations on accuracy values with a single execution, but rather to base them on K-fold directly, so as not to have inefficient results in the end as happened with the ANN.

It must also be considered that the experiment could last 10 times longer or many more computational resources, so it must be assessed how necessary this implementation would be, given the results already obtained. In general, if this research is compared with those generated by the authors of the related work, it can be concluded that competent results were achieved comparatively to the best results of each work, even better in most cases.

However, it must be mentioned that they have not been compared with the same databases, so this comparison, for the moment, would only be in terms of accuracy, which is incomplete and would require, in the future, comparison with the same database of each author to be able to define it.

It can also be concluded that, with only the financial ratios of a company, its financial bankruptcy can be predicted. Particularly, with the 95 financial ratios used in this work. In future work, other models for predicting financial bankruptcy will be considered, like LSTM network, recurrent network, variants of neuro-fuzzy networks, etc.

References

1. **Abdou, H., Abdallah, W., Mulkeen, J., Ntim, C. G., Wang, Y. (2017).** Prediction of financial strength ratings using machine learning and conventional techniques. Investment Management and Financial Innovations, Vol. 14, No. 4, pp. 194–211.
2. **Arieshanti, I., Purwananto, Y., Ramadhani, A., Nuha, M. U., Ulinnuha, N. (2013).** Comparative study of bankruptcy prediction models. TELKOMNIKA (Telecommunication Computing Electronics and Control), Vol. 11, No. 3, pp. 591. DOI: 10.12928/telkomnika. v11i3.1143.
3. **Arora, N. Saini, J. R. (2013).** Time series model for bankruptcy prediction via adaptive neuro-fuzzy inference system. International Journal of Hybrid Information Technology, Vol. 6, No. 2, pp. 51–64.
4. **Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. (2002).** Smote: synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, Vol. 16, pp. 321–357. DOI: 10.1613/jair.953.
5. **Chen, Y., Liu, K., Xie, Y., Hu, M. (2020).** Financial trading strategy system based on machine learning. Mathematical Problems in Engineering, Vol. 2020, pp. 1–13. DOI: 10.11 55/2020/3589198.
6. **Cunningham, P., Delany, S. J. (2021).** K-nearest neighbour classifiers - A tutorial. ACM Computing Surveys, Vol. 54, No. 6, pp. 1–25. DOI: 10.1145/3459665.
7. **Dantine, J. P., Donaldson, J. B. (2014).** Intermediate financial theory. Third edition, Academic Press.
8. **Ekman, M. (2021).** Learning deep learning: Theory and practice of neural networks computer vision, natural language processing, and transformers using tensorflow. Addison-Wesley Professional.
9. **Gad, A. G. (2022).** Particle swarm optimization algorithm and its applications: A systematic review. Archives of Computational Methods in Engineering, Vol. 29, No. 5, pp. 2531–2561. DOI: 10.1007/s11831-021-09694-4.
10. **Gaxiola, F., Melin, P., Valdez, F., Castro, J. R. (2017).** Optimization of deep neural network for recognition with human iris biometric measure. Advances in Intelligent Systems and Computing, pp. 172–180. DOI: 10.1007/978-3-319-67137-6_19.
11. **Gaxiola, F., Melin, P., Valdez, F., Castro, J. R. (2016).** Optimization of type-2 and type-1 fuzzy integrator to ensemble neural network with fuzzy weights adjustment. Studies in Computational Intelligence, pp. 39–61. DOI: 10.1007/978-3-319-47054-2_3.
12. **Goodfellow, I., Bengio, Y., Courville, A. (2016).** Convolutional networks. Lecture slides for Chapter 9 of Deep Learning.

13. **Guajardo, G., Andrade, N. (2012).** Estados financieros. Contabilidad para no contadores.
14. **Hamdia, K. M., Zhuang, X., Rabczuk, T. (2020).** An efficient optimization approach for designing machine learning models based on genetic algorithm. Neural Computing and Applications, Vol. 33, No. 6, pp. 1923–1933. DOI: 10.1007/s00521-020-05035-x.
15. **Houssein, E. H., Gad, A. G., Hussain, K., Suganthan, P. N. (2021).** Major advances in particle swarm optimization: theory, analysis, and application. Swarm and Evolutionary Computation, Vol. 63, pp. 100868. DOI: 10.1016/j.swevo.2021.100868.
16. **Jang, J. S. R., Sun, C. T., Mizutani, E. (1997).** Neuro-fuzzy and soft computing-a computational approach to learning and machine intelligence. IEEE Transactions on automatic control, Vol. 42, No. 10, pp. 1482-1484.
17. **Jang, J. (1993).** ANFIS: adaptive-network-based fuzzy inference system. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23, No. 3, pp. 665–685. DOI: 10.1109/21.256541.
18. **Kansal, A. K., Sharma, S. (2019).** A methodological review of financial distress prediction techniques. 4th International Conference on Advances in Management & Digital Sciences, Vol. 1, pp. 294–301.
19. **Karaboga, D., Kaya, E. (2019).** Adaptive network based fuzzy inference system (ANFIS) training approaches: A comprehensive survey. Artificial Intelligence Review, Vol. 52, No. 4, pp. 2263–2293. DOI: 10.1007/s10462-017-9610-2.
20. **Liang, D., Tsai, C. F. (2020).** Taiwanese bankruptcy prediction. UCI machine learning repository. https://archive.ics.uci.edu/ml/data sets/Taiwanese+Bankruptcy+Prediction.
21. **López, J. F. (2019).** Bankruptcy. Economipedia.com. https://economipedia. com/definiciones/quiebra.html#:~:text=Una%2 0quiebra%20o%20bancarrota%20es,su%20a ctividad%20de%20forma%20permanente.
22. **Mselmi, N., Lahiani, A., Hamza, T. (2017).** Financial distress prediction: the case of french small and medium-sized firms. International Review of Financial Analysis, Vol. 50, pp. 67–80. DOI: 10.1016/j.irfa.2017.02.004.
23. **Muslim, M. A., Dasril, Y. (2021).** Company bankruptcy prediction framework based on the most influential features using XGBoost and stacking ensemble learning. International Journal of Electrical and Computer Engineering (IJECE), Vol. 11, No. 6, pp. 5549. DOI: 10.11591/ijece.v11i6.pp5549-5557.
24. **Narvekar, A., Guha, D. (2021).** Bankruptcy prediction using machine learning and an application to the case of the COVID-19 recession. Data Science in Finance and Economics, Vol. 1, No. 2, pp. 180–195. DOI: 10.3934/dsfe.2021010.
25. **Park, M. S., Son, H., Hyun, C., Hwang, H. J. (2021).** Explainability of machine learning models for bankruptcy prediction. IEEE Access, Vol. 9, pp. 124887–124899. DOI: 10. 1109/access.2021.3110270.
26. **Pisner, D. A., Schnyer, D. M. (2020).** Support vector machine. Machine Learning, pp. 101–121. DOI: 10.1016/b978-0-12-815739-8.00006-7.
27. **Santoso, N., Wibowo, W. (2018).** Financial distress prediction using linear discriminant analysis and support vector machine. Journal of Physics: Conference Series, Vol. 979, pp. 012089. DOI: 10.1088/1742-6596/979/ 1/012089.
28. **Shetty, S., Musa, M., Brédart, X. (2022).** Bankruptcy prediction using machine learning techniques. Journal of Risk and Financial Management, Vol. 15, No. 1, pp. 35. DOI: 10. 3390/jrfm15010035.
29. **Stasko, A., Birzniece, I., Kebers, G. (2021).** Development of bankruptcy prediction model for Latvian companies. Complex Systems Informatics and Modeling Quarterly, No. 27, pp. 45–59. DOI: 10.7250/csimq.2021-27.02.
30. **Sudarsanam, S. K. (2016).** A fuzzy neural network model for bankruptcy prediction. Journal of Engineering Computers & Applied Sciences (JECAS), Vol. 5, No. 6, pp. 33–40.
31. **Suryawanshi, R., Takavane, T., Shah, T., Mathur, R. (2022).** Stocks and cryptocurrency price prediction using long short term memory.

International Journal of Engineering Research & Technology, Vol. 11, No. 5, pp. 560–564.

32. **Tabbakh, A. (2021).** Bankruptcy prediction using robust machine learning model. Turkish Journal of Computer and Mathematics Education, Vol. 12, No. 10, pp. 3060-3073.
33. **Taunk, K., De, S., Verma, S., Swetapadma, A. (2019).** A brief review of nearest neighbor algorithm for learning and classification. International Conference on Intelligent Computing and Control Systems, pp 1255–1260. DOI: 10.1109/iccs45141.2019.9065747.
34. **Vera-García, I. (2017).** El modelo Z de Altman como herramienta financiera para pronosticar o predecir el desempeño financiero de las empresas mexicanas cotizadas. Caso de las empresas manufactureras del sector alimenticio. Master degree thesis, Universidad Autonoma del Estado de Hidalgo, UAEH Digital Library. http://dgsa.uaeh.edu.mx:8080/bibliotecadigital/handle/231104/2381.
35. **Wang, H., Liu, X. (2021).** Undersampling bankruptcy prediction: Taiwan bankruptcy data. PLOS ONE, Vol. 16, No. 7, pp. e0254030. DOI: 10.1371/journal.pone.0254030.
36. **Wang, L. (2005).** Support vector machines: theory and applications. Singapur: Springer. Vol 177.
37. **Wang, N. (2017).** Bankruptcy prediction using machine learning. Journal of Mathematical Finance, Vol. 07, No. 04, pp. 908–918. DOI: 10.4236/jmf.2017.74049.
38. **Wu, J. (2020).** Essentials of pattern recognition: An accessible approach. Cambridge University Press.
39. **Xie, C., Luo, C., Yu, X. (2010).** Financial distress prediction based on SVM and MDA methods: the case of chinese listed companies. Quality & Quantity, Vol. 45, No. 3, pp. 671–686. DOI: 10.1007/s11135-010-9376- y.
40. **Yang, X. S. (2020).** Nature-inspired optimization algorithms. Academic Press.
41. **Zhang, A., Lipton, Z. C., Li, M., Smola, A. J. (2021).** Dive into deep learning. ArXiv. DOI: 10.48550/ARXIV.2106.11342.

Article received on 31/01/2024; accepted on 12/03/2024.
**Corresponding author is Fernando-Gaxiola.*

ISSN 2007-9737

Dragonfly Algorithm for Benchmark Mathematical Functions Optimization

Hector M. Guajardo, Fevrier Valdez*

Tecnológico Nacional de México, Campus Tijuana, Mexico

fevrier@tectijuana.mx

Abstract. In this paper, we study the dragonfly algorithm, an optimization method derived from the observation of nature and mathematical modeled after the swarming behavior of dragonflies. Xin-She Yang devised this approach, which has been applied to various optimization challenges. [1]. The algorithm effectively explores the search space by imitating dragonfly behaviors like hunting for prey, fleeing from predators, and swarming. The method consists in apply Type-1 Fuzzy Logic to some of the parameters of the algorithm, in this case, W and Betha to analyze the results applied to the mathematical functions F1 through F10 included in this paper, once that we have applied the adaptation of parameters, we will review the results compared with the rest of the papers that implement the same mathematical functions, so we can have a general idea if this method can be reliable.

Keywords: Optimization, dragonfly algorithm, bio-inspired, type-1 fuzzy logic.

1 Introduction

Search and optimization algorithms such as Particle Swarm Optimization (PSO), Differential Evolution (DE), Genetic Algorithm (GA), Firefly Algorithm (FA), and Dragonfly Algorithm (DA) have proven to be efficient in terms of speed and convergence for certain types of problems, so we expect the optimization algorithm to work efficiently applied to intelligent computing optimization problems. These algorithms are based on bioinspired principles and have been extensively studied and tested in scientific literature.

The choice of a specific algorithm depends on the optimization problem in question and its characteristics. However, combining different search and optimization algorithms in an algorithm can further improve efficiency in terms of convergence speed and the quality of the solution obtained. Algorithms that combine different search and optimization strategies can take advantage of the strengths of each of the algorithms and overcome their limitations.

In summary, an optimization algorithm combining PSO, DE, GA, and FA are expected to perform efficiently when applied to optimization problems due to its proven effectiveness in terms of speed of convergence in solving optimization problems. Where there are insects like dragonflies, fireflies, and damselflies, there are many more insects with the same similarities for hunting, reproduction, or in matters of movement on the entire ecosystem.

Observing the behaviors and structures of organisms in nature often suggests they perform their functions exceptionally well. Numerous studies have been conducted on this subject, including one by Xin-She Yang that began with publications pertaining to the firefly algorithm. The classification of many species of related insects will be reviewed in the following study [2].

Metaheuristics encompass broad strategies that skillfully blend different methodologies to navigate through the solution space. In design of optimization, the design objective can be as simple as maximizing production efficiency or minimizing production costs. An optimization algorithm is a technique that compares several solutions repeatedly until an ideal or feasible answer is identified. Currently, two types of optimization methods are frequently employed.

To transition from one solution to the next, deterministic algorithms employ a set of rules.

Fig. 1. The dragonfly cycle represents the steps the dragonfly must go through to become an adult dragonfly

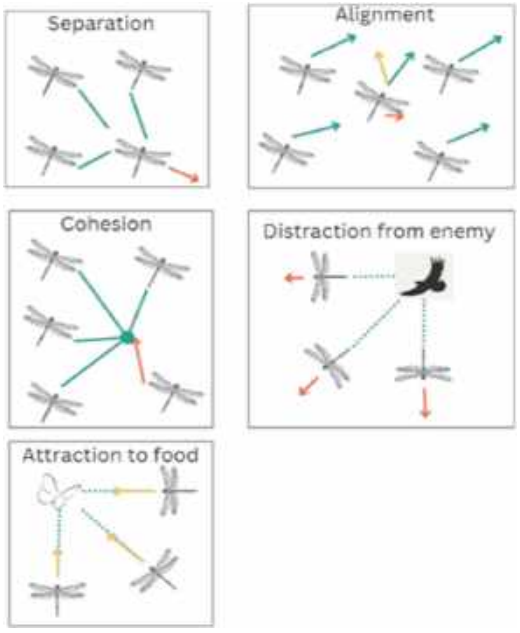


Fig. 2. The dragonfly patterns between individuals in the swarm

These algorithms have been effectively employed to solve a variety of engineering design challenges [3].

Since stochastic algorithms include probabilistic translation rules and a random nature, they may execute in a different sequence or produce a different outcome each time they are run with the same input.

The layout of this article is outlined as follows: 1. Introduction, where we provide a summary of the article's content; 2. Nature Inspiration the new optimization technique we offer is based on organic inspiration; 3. Literature review, which includes all the studies relevant to the topics of this article; 4. The presentation and implementation of the Dragonfly Algorithm (DA); 5. Type-1 fuzzy logic explanation, 6. Results, analysis, and comparison of the experiments performed; 7. Analytical conclusions and a summary of the investigation are described in this article.

2 Nature Inspiration

In some cultures, dragonflies were called kachi-mushi (victorious insects) because they only fly forward, which gave them the character of those who never retreat and always move forward, whatever the circumstances.

Thus, it became a symbol of strength, courage and determination. About 5,000 species of dragonflies are known and the algorithm is inspired by static behavior and dynamic behavior all dragonflies can move together to the same place for example a migration.

In the following, we can review the life cycle of the dragonfly.

The adult dragonfly starts to mate, after that, they will lay their eggs in or around the water eggs in the water, after the eggs will become a larva and after that the dragonfly will emerge. This can be appreciated in Fig 1.

Based on these factors, it was suggested an algorithm that follows the model of insects and how they adjust to physiological changes by sharing resources and communicating to survive and grow. This algorithm was called "dragonfly".

Natural optimization methods have proven to be adaptable, flexible, and efficient in handling practical problems.

Algorithm 1 Dragonfly algorithm pseudocode

Initialize Dragonfly population randomly
Initialize Step vector/Size for dragonfly

While (current iteration < maximum iteration)
  Calculate fitness values for each dragonfly.
  Update food sources and enemy.
  Update parameters w, s, a, c, f and e.
  Calculate S, A, C and F.
  Update the neighboring radius.

  if dragonfly has at least one neighboring dragonfly.
    Update velocity and position.
  else
    Update position vector
  else if

  Check and correct the new positions based on the boundaries of variables.
End While

The fact that there is currently no optimization technique that can handle all problems is well recognized [5]. There are everyday issues that can be solved in business, economics, research, and other fields. There is a wide variety of optimization techniques, and some are more effective than others in solving specific problems. There are various metaheuristic optimization techniques, for example, Particle Swarm Optimization (PSO) [10], Artificial Colony Optimization (ACO) [8, 11], Artificial Bee Algorithm (ABC) [9], Firefly Algorithm (FA) [12], as well as other algorithms based on Hill climbing swarms [13], genetic algorithms (GA) [6], and different techniques based on trajectories. Differential evolution (DE) [14] and genetic programming (GP) [16] are examples of evolutionary algorithms.

3 Study of Literature

There are several social behaviors used in nature to carry out various tasks. Although survival is the goal of all individuals and collective actions, organisms collaborate and interact in groups for a variety of purposes, including hunting, defending, navigating, and foraging.

Wolf packs, for example, have some of the best-structured social interactions for hunting. Wolves often follow a social hierarchy to pursue prey in various ways: chasing, circling, tormenting, and attacking. [17]. Holland authored a book detailing the development of genetic algorithms (GAs). De Jong concluded his research by showcasing the considerable potential and robustness of evolutionary algorithms across various objective functions, including those that are noisy, multimodal, or discontinuous [6]. To minimize their learning and prediction errors through iterative trial and error, artificial neural networks, support vector machines, and other machine learning approaches are genetic algorithms and can be considered a heuristic optimization methodology.

In 1961 Van Bergeijk, W. A, Harmon, L. D. and Levinson, J. Z., and Harmon, L. D. proposed artificial neurons as simple information processing units [19, 20, 21].

Particle Swarm Optimization (PSO), an optimization method inspired by the collective intelligence of fish, birds, and even human beings, was created in 1995 by James Kennedy and Russell C. Eberhart [22, 23].

Table 1. Description of the algorithm parameters

Parameter	Description	Values
Population	Population size	40
Boundaries	Number of boundaries	2
Dimensions	Dimension's size	8, 10, 16, 32
Iterations	Iterations size	500
S	Separation weight	
A	Alignment weight	
C	Cohesion	
F	Food factor	
E	Enemy factor	
w	Inertia weight	
S_i	Separation of the i=th individual	
A_i	Alignment of the i=th individual	
C_i	Cohesion of the i=th individual	
F_i	Food source of the i=th individual	
E_i	Enemy position of the i=th individual	
t	Current iteration	

Table 2. CEC2013 math functions

Fun	Name	Range	Nature
F1	Sphere	[-5.12, 5.12]	U
F2	Rosenbrock	[-5, 10]	U
F3	Griewank	[-600, 600]	M
F4	Rastrigin	[-5.12, 5.12]	M
F5	Ackley	[-32.768, 32.768]	M

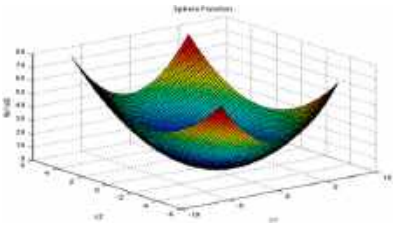
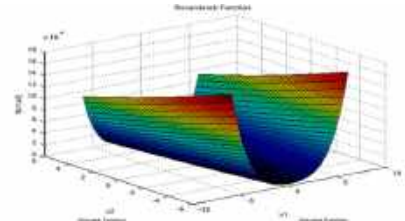
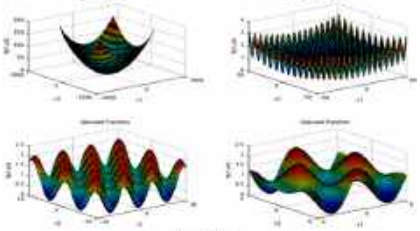
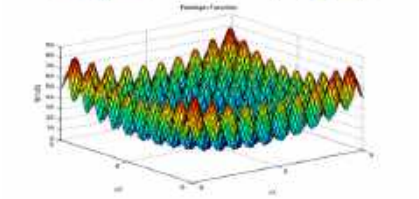
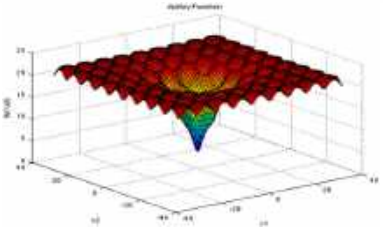
Fun	Graph	Equation
1		$f(\mathbf{x}) = \sum_{i=1}^{d} x_i^2$
2		$f(\boldsymbol{x}) = \sum_{i=1}^{d-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$
3		$f(x) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
4		$f(x) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)]$
5		$f(x) = -a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sqrt{\sum_{i=1}^{d} \cos(cx_i)}\right) + a + \exp(1)$

Fig. 4. Plots of five CEC2013 mathematical functions

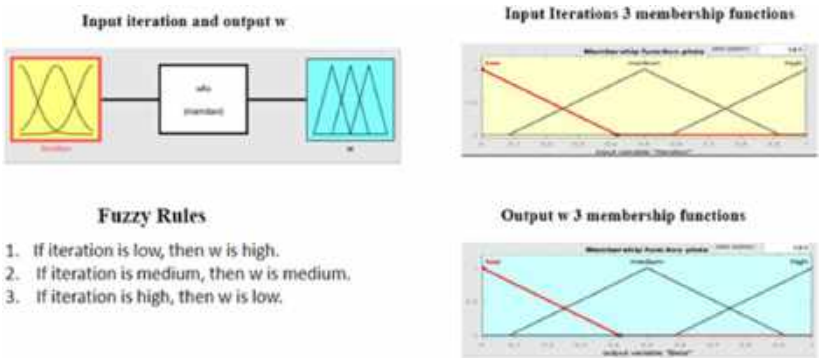


Fig. 5 Type-1 fuzzy inference systems w parameter

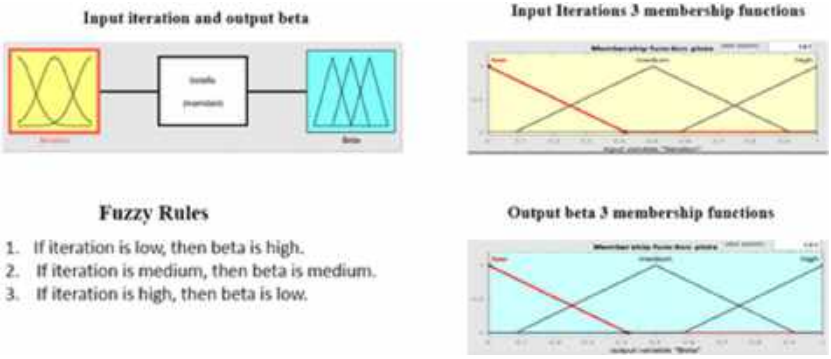


Fig. 6 Type-1 fuzzy inference systems beta parameter

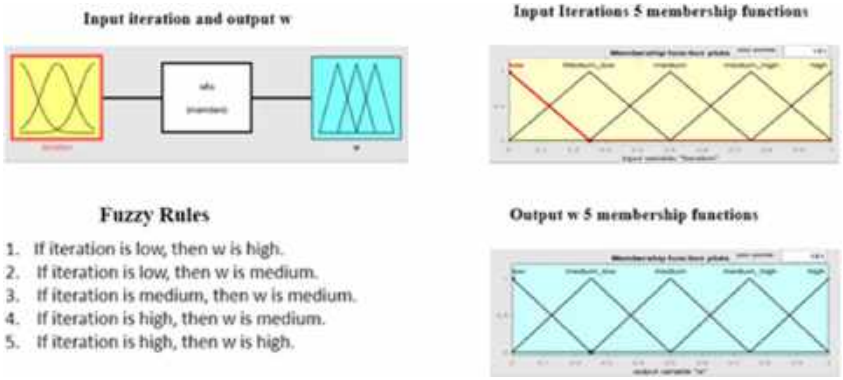


Fig. 7. Type-1 fuzzy inference systems beta parameter

With time, the PSO method has demonstrated its superiority over conventional algorithms and genetic algorithms in specific problem domains, though it may not be suitable for every scenario. There isn't a universal algorithm that excels in all optimization problems; hence, current research aims to identify the most effective and efficient algorithm(s) for particular tasks. D. H. Wolpert and W. G.

Macready introduced the No-Free Lunch theorems to caution the scientific community that if algorithm A outperforms algorithm B for certain optimization functions, then B is likely to outperform A for other functions. [24]. Over time, researchers S. Nakrani and C. Tovey suggested the honey bee algorithm and its use as a foraging algorithm for problems including multimodal and dynamic optimization. [25].

Table 3. Comparison results for 30 dimensions of CMOA, DA and DA with Type-1

Fun	DA		CMOA		DA Type-1	
	Mean	Std dev	Mean	Std dev	Mean	Std dev
F1	2.85E-18	7.16E-01	8.11E-09	4.79E-09	1.25E+00	2.88E+0
F2	7.60E+00	6.79E+0	6.58E-09	3.32E-09	4.41E-01	4.79E-01
F3	1.03E-02	4.69E-03	1.64E-09	1.39E-09	1.00E+01	5.15E+01
F4	1.60E+01	9.48E+0	8.13E-09	4.42E-09	1.25E+00	1.55E+0
F5	2.31E-01	4.87E-01	1.44E-09	1.21E-09	2.11E+02	3.21E+02

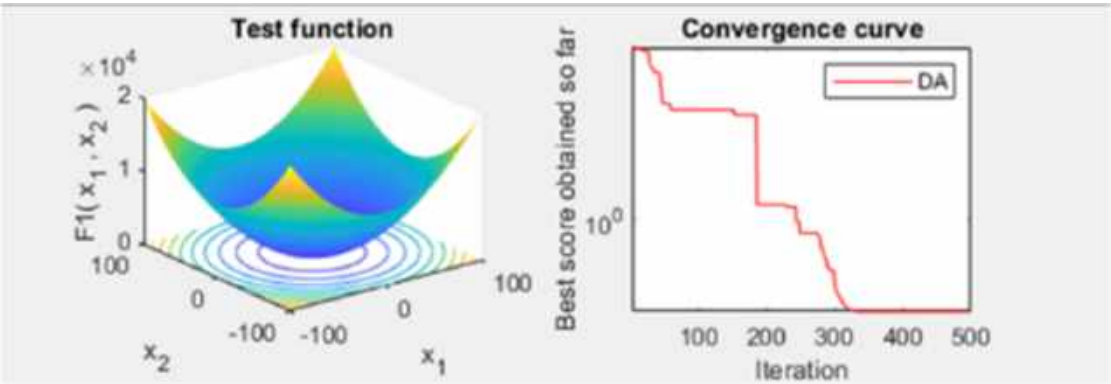


Fig. 8. Display and convergence for function 1

Table 4. Results for function 1 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	1.25E+00	4.09E+00	1.30E+02	1.53E+03
Std	2.88E+00	1.19E+01	1.70E+02	6.96E+02

The techniques were inspired by how actual bees feed in the nature. To identify spreaders utilizing a variety of targets, Amir S. and Ahman Z. developed the artificial bee colony (ABC) algorithm in 2020. [9]. Particle Swarm Optimization (PSO) and the Fire-fly Algorithm (FA), inspired by the flashing patterns of fireflies, were combined in a practical project undertaken by Khennak, I., Drias, H., and Drias, Y. [12].

Seyedali Mirjalili et al. proposed the Grey Wolf Optimizer (GWO) in 2013 [25], Inspired by grey wolves (Canis lupus), it imitates the natural leadership structure and hunting strategy of these canines. The Coyote Optimization Algorithm, which Juliano Pierezan and Leandro dos Santos Coelho created in 2018, is a population-based metaheuristic for optimization that draws inspiration from the canis latrans species [26].

In 2020, Abdolkarim Mohammadi-Balani and his team introduced their innovative Golden Eagle Optimizer algorithm, designed to adjust speed at different points along a spiral trajectory, mimicking the hunting behavior of golden eagles. [27].

Additionally, new metaheuristic algorithms that are better than others at solving a particular kind of problem will continue to be developed.

4 Dragonfly Algorithm (DA)

In nature, practically all other little insects are preyed upon by dragonflies, which are thought of as small predators. Additionally, nymph dragonflies eat other maritime insects and even small fish. The intriguing characteristic of dragonflies is their uncommon and unusual swarming behavior.

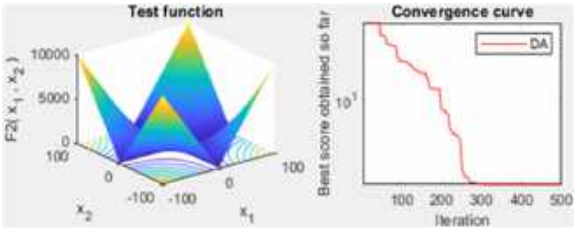


Fig. 9. Display and convergence for function 2

Table 5. Results for function 2 experiments

500 Iterations				
40 Dragonflies				
Exp	**8 Dim**	**10 Dim**	**16 Dim**	**32 Dim**
Ave	4.43E-01	1.00E-07	3.79E+00	1.48E+01
Std	4.71E-01	1.00E-07	1.55E+00	7.64E+00

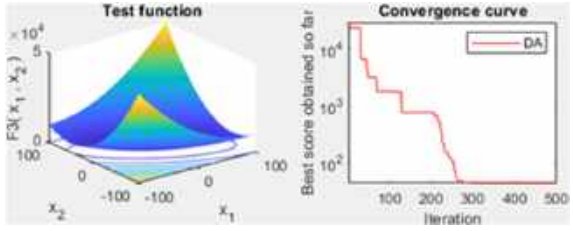


Fig. 10. Display and convergence for function 3

Table 6. Results for function 3 experiments

500 Iterations				
40 Dragonflies				
Exp	**8 Dim**	**10 Dim**	**16 Dim**	**32 Dim**
Ave	2.69E+01	1.80E+02	1.46E+03	1.30E+04
Std	5.08E+01	2.43E+02	1.70E+03	9.96E+03

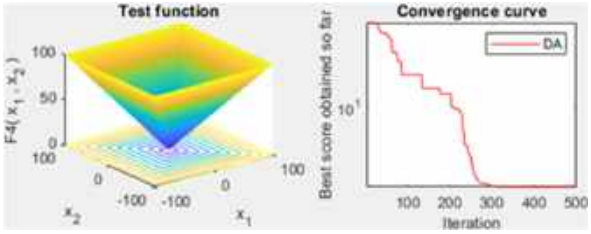


Fig. 11. Display and convergence for function 4

Table 7. Results for function 4 experiments

500 Iterations				
40 Dragonflies				
Exp	**8 Dim**	**10 Dim**	**16 Dim**	**32 Dim**
Ave	1.24E+00	1.85E+00	9.17E+00	2.93E+01
Std	1.52E+00	1.36E+00	5.81E+00	1.06E+01

Only two things cause dragonflies to swarm: migration and hunting. Both are referred to as swarms—the former as a static (feeding) swarm and the latter as a dynamic (migratory) swarm.

As already mentioned, the DA is an optimization technique that draws inspiration from the same-named bug [28]. The static and dynamic characteristics of swarms serve as the primary source of inspiration for the DA algorithm.

These two are extremely like the exploration and exploitation phases of metaheuristic optimization. The main goal of the exploration phase is for dragonflies to organize into sub-swarms and fly in a static swarm over numerous locations.

Certainly, during times of static swarming, it is observed that dragonflies tend to fly together in larger groups, all aligning their flight paths—a behavior that is notably advantageous, particularly during the exploitation phase.

The primary goal of any swarm is survival, every member should be drawn to food sources and vigilant against external threats.

As demonstrated on Fig. 2 the five essential factors that affect how individuals in swarms update their positions considering these two behaviors. Swarm behavior follows three important principles:

1. Separation: Individual avoid static collision with neighbor:

$$S_j = \sum_{j=1}^{N} X - X_j. \tag{1}$$

2. Alignment: Individual velocity matched with neighbor individuals:

$$A_i = \sum_{j=1}^{N} V_j. \tag{2}$$

3. Cohesion: Individual tendency toward center of the herd:

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X. \tag{3}$$

X: This represents the position of an individual, which typically denotes its current location or coordinates in each space.

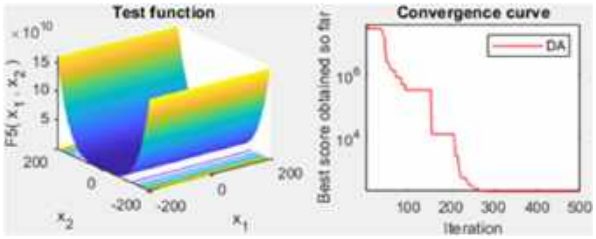


Fig. 12. Display and convergence for function 5

Table 8. Results for function 5 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	2.05E+02	3.51E+03	1.25E+04	1.88E+05
Std	3.17E+02	1.67E+04	2.86E+04	1.88E+05

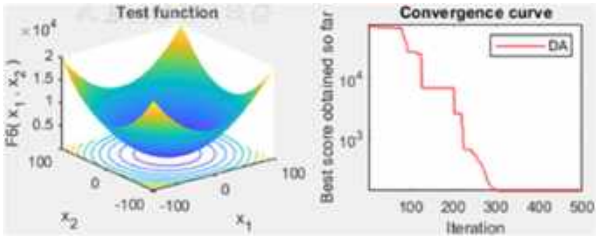


Fig. 13. Display and convergence for function 6

Table 9. Results for function 6 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	1.25E+00	2.76E+00	7.44E+01	1.50E+03
Std	5.38E+00	4.87E+00	1.17E+02	1.22E+03

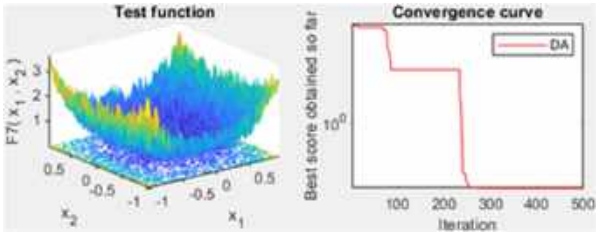


Fig. 14. Display and convergence for function 7

Table 10. Results for function 7 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	1.39E-02	2.73E-02	6.21E-02	5.83E-01
Std	1.13E-02	2.13E-02	5.26E-02	3.27E-01

V_j: This represents the velocity of an individual, which typically refers to the speed and direction at which it is moving.

N: This represents the number of neighborhoods or groups of individuals in your system. It is essentially a parameter that determines how individuals are grouped or organized into neighborhoods. Attraction to food source is calculated:

$$F_i = X^+ - X, \tag{4}$$

where:

X : Is the position of the current individual,

X^+: Is the area of the food.

Distraction from enemy is calculated:

$$E_i = X^- + X, \tag{5}$$

where:

X : Is the position of the current individual,

X^-: Is the area of the enemy.

In this research, dragonfly behavior is supposed to be a combination of these five corrective patterns. Two vectors are used to update the position of artificial dragonflies in a search space and replicate their movements: step (ΔX) and position (X). The step vector represents the direction of the dragonfly movement and is described as follows:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + w\Delta X_i, \tag{6}$$

where s is the separation weight S_i representing $i = th$ individual's separation, a represents the alignment weight, A represents $i = th$ the individual's alignment, and c represents the cohesion weight. C_i is the $i = th$ individual's cohesiveness, f is the food factor, and F_i is the $i = th$ individual's food supply, e is the enemy factor, E_i is the $i = th$ individual's position of enemy, w is the inertia weight, and t is the iteration timer, in Algorithm 1 we can see with more detail the pseudocode from dragonfly algorithm.

Table 1 presents the names and concise descriptions of all the parameters applied in the Dragonfly algorithm. In the subsequent section of this paper, we provide the outcomes of the experimentation conducted in this study.

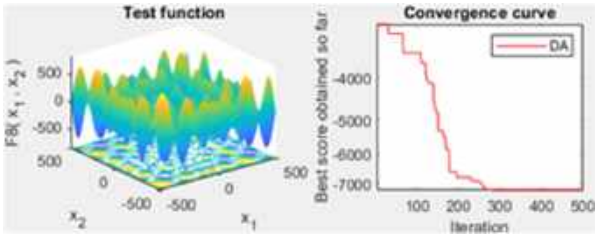


Fig. 15. Display and convergence for function 8

Table 11. Results for function 8 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	-2.48E+03	-2.89E+03	-3.78E+03	-5.78E+03
Std	3.47E+02	3.19E+02	4.74E+02	7.51E+02

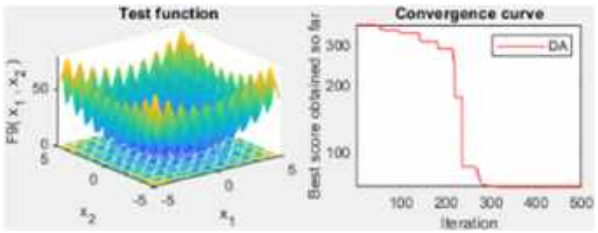


Fig. 16. Display and convergence for function 9

Table 12. Results for function 9 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	1.86E+01	2.77E+01	5.79E+01	1.56E+02
Std	1.03E+01	1.29E+01	2.05E+01	3.53E+01

Table 2 outlines the titles of five mathematical functions from the CEC2013 dataset, along with their corresponding scopes and characteristics, which were utilized in our investigation. The Fig. 4 presents graphical representations and mathematical equations for five out of the ten mathematical functions utilized in this study.

5 Type-1 Fuzzy Logic

Fuzzy logic, alternatively referred to as fuzzy sets theory, provides a mathematical framework for addressing reasoning and decision-making in scenarios characterized by uncertainty and imprecision. Unlike traditional binary logic where statements are either true or false, fuzzy logic allows for degrees of truth between 0 and 1, representing degrees of membership or truthfulness.

This allows for more nuanced modeling and analysis, particularly in areas where precise boundaries are difficult to define. [29] Key Concepts of Fuzzy Logic: Fuzzy Sets: Fuzzy sets are a fundamental concept in fuzzy logic, introduced by Lotfi Zadeh in 1965. Unlike classical sets where an element either belongs to a set or does not, fuzzy sets allow for degrees of membership.

In a fuzzy set, each element has a membership value that represents the degree to which the element belongs to the set. These membership values range between 0 and 1, where 0 indicates no membership, 1 indicates full membership, and values in between represent degrees of partial membership.

Fuzzy sets are especially useful for modeling uncertainties and vagueness present in many real-world systems. Membership Functions: Membership functions are mathematical functions that define the degree of membership of each element in a fuzzy set.

These functions map each element from the universal set to a real number in the interval [0, 1]. There are various types of membership functions, such as triangular, trapezoidal, Gaussian, and sigmoidal, each suited for different applications and interpretations.

The choice of membership function depends on the specific characteristics of the problem domain and the preferences of the modeler. Membership functions play a crucial role in fuzzy logic systems as they determine the degree to which fuzzy sets represent real-world phenomena. Fuzzy Operators: Fuzzy operators are mathematical operations defined on fuzzy sets that allow for combining and manipulating fuzzy information.

These operators extend classical set operations such as union, intersection, and complement to accommodate the degrees of membership associated with fuzzy sets. The basic fuzzy operators include union (OR), intersection (AND), and complement (NOT). Additionally, there are other operators like algebraic product, bounded sum, and drastic sum, each serving different purposes in fuzzy logic systems.

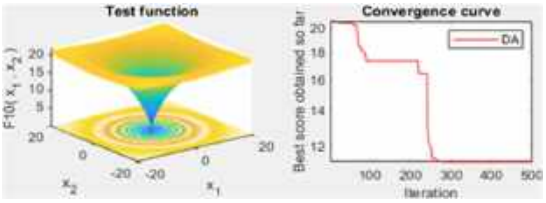


Fig. 17. Display and convergence for function 10

Table 13. Results for function 10 experiments

500 Iterations				
40 Dragonflies				
Exp	8 Dim	10 Dim	16 Dim	32 Dim
Ave	1.52E+00	2.57E+00	4.42E+00	9.53E+00
Std	1.14E+00	1.54E+00	1.41E+00	1.72E+00

Fuzzy operators are essential for performing reasoning and making decisions in fuzzy logic-based control systems, pattern recognition, and other applications. [32, 33, 34]. Example: Consider the concept of "temperature" in a room. Instead of categorizing it simply as "hot" or "cold," fuzzy logic allows for a more nuanced approach.

We might define a fuzzy set for "comfortable temperature" with a membership function that peaks around 22 degrees Celsius. Then, if the room is at 20 degrees, it might have a membership value of 0.8 in the "comfortable temperature" set, indicating it is somewhat comfortable but not perfect.

Adaptation of Type-1 Fuzzy Logic

We will present Type-1 fuzzy inference systems implemented for the parameters w and beta of the Dragonfly algorithm in the following table. Here, in Fig. 5, we can observe the inference functions with the iteration parameter as input and the parameter was output, with their three fuzzy rules:

Fuzzy Rules for the Parameter w:

1. If iteration is low, then w is high.
2. If iteration is medium, then w is medium.
3. If iteration is high, then w is low.

Here, in Fig. 6 we can observe the inference functions with the iteration parameter as input and the beta parameter as output, with their three fuzzy rules.

Fuzzy Rules for the Parameter Beta:

1. If iteration is low, then beta is high.
2. If iteration is medium, then beta is medium.
3. If iteration is high, then beta is low.

In Fig. 7, we can observe the inference functions with the iteration parameter as input and the parameter was output, with its five fuzzy rules:

Fuzzy Rules for the Parameter w:

1. If iteration is low, then w is high.
2. If iteration is low, then w is medium.
3. If iteration is medium, then w is medium.
4. If iteration is high, then w is medium.
5. If iteration is high, then w is high.

6 Results and Comparison

In Table 3, we present a comparison of outcomes derived from two distinct algorithms: The Dragonfly Algorithm (DA), where the Dragonfly algorithm is integrated with type-1 application to parameters such as Beta and w, and the Continuous Mycorrhiza Optimization Algorithm (CMOA). The table showcases the most favorable mean values and standard deviations obtained from each experiment across different functions.

The bio-inspired Dragonfly Algorithm serves as a remarkable example of how nature's mechanisms can provide creative and effective solutions to modern technological obstacles. Table 4 showcases the best results acquired from 30 experiments conducted across dimensions of 8, 10, 16, and 32.

These experiments employed a population of 40 dragonflies and a maximum of 500 iterations, tailored specifically for Function 1. Fig 8 depicts the graphical representation and convergence curve for Function 1.

Table 5 displays the best results achieved from 30 experiments carried out across dimensions of 8, 10, 16, and 32. These experiments employed a population of 40 dragonflies and a maximum of 500 iterations, all tailored for Function 2. Fig 9 visually

represents the performance and convergence curve for Function 2.

Table 6 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32.

These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 3. In Fig. 10, it is illustrated the display and convergence curve for Function 3. Table 7 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32.

These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 4. In Fig. 11, we illustrate the display and convergence curve for Function 4.

Table 8 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32. These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 5. In Fig. 12, we illustrate the display and convergence curve for Function 5.

Table 9 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32. These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 6.

In Fig. 13, it is illustrated the display and convergence curve for Function 6. Table 10 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32. These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 7.

In Fig. 14, we illustrate the display and convergence curve for Function 7. Table 11 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32. These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 8.

In Fig. 15, we illustrate the display and convergence curve for Function 8. Table 12 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32.

These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 9. In Fig. 16, it is illustrated the display and convergence curve for Function 9.

Table 13 presents the optimal values obtained from 30 experiments conducted across dimensions of 8, 10, 16, and 32. These experiments utilized a population of 40 dragonflies and a maximum number of 500 iterations, all specifically designed for Function 10. Fig. 17 illustrates the display and convergence curve for Function 10.

7 Conclusions

In conclusion, when comparing the Dragonfly method with itself, utilizing a population of 40 Dragonflies and an iteration value of 500, but with the incorporation of Type-1 fuzzy logic adaptation for parameters w and beta, we can confidently state that much better results are obtained.

Furthermore, when compared to the CMOA algorithm in 50 dimensions, the Dragonfly method continues to yield superior outcomes, as evidenced in more detail in Table 3, significantly enhancing the results when mean and standard deviation are applied across all experiments conducted.

We will continue to generate results, with future work focusing on adapting Type-2 fuzzy logic to further compare outcomes with other algorithms. The findings will be shared with the community to support fellow researchers.

It is worth mentioning that while the DA method may not represent the ultimate optimization technique presently accessible, it does demonstrate potential and can be beneficial in particular optimization problem contexts.

As future work, we will consider other metaheuristics for the same approach, like in [35-40]. Later, we expect to utilize type-2 fuzzy logic

[41-44] for parameter adaptation in the dragon fly algorithm, as in [45-46].

References

1. **Yang, X. (2010).** A new metaheuristic bat-inspired algorithm. Studies in Computational Intelligence, Vol. 284, pp. 65–74. DOI: 10.1007/978-3-642-12538-6_6.
2. **Bybee, S. M., Kalkman, V. J., Erickson, R. J., Frandsen, P. B., Breinholt, J. W., Suvorov, A., Dijkstra, K. B., Cordero-Rivera, A., Skevington, J. H., Abbott, J. C., Herrera, M. S., Lemmon, A. R., Lemmon, E. M., Ware, J. L. (2021).** Phylogeny and classification of odonata using targeted genomics. Molecular Phylogenetics and Evolution, Vol. 160, pp. 107115. DOI: 10.1016/j.ympev.2021.107115.
3. **Osman, I. H., Kelly, J. P. (1996).** Meta-heuristics: An overview. Meta-Heuristics, pp. 1–21. DOI: 10.1007/978-1-4613-1361-8_1.
4. **Glover, F., Mulvey, J. M., Hoyland, K. (1996).** Solving dynamic stochastic control problems in finance using tabu search with variable scaling. Meta-Heuristics, pp. 429–448. DOI: 10.1007/978-1-4613-1361-8_26.
5. **Carreres-Prieto, D., Ybarra-Moreno, J., García, J. T., Cerdán-Cartagena, J. F. (2023).** A comparative analysis of neural networks and genetic algorithms to characterize wastewater from led spectrophotometry. Journal of Environmental Chemical Engineering, Vol. 11, No. 3, pp. 110219. DOI: 10.1016/j.jece.2023.110219.
6. **Holland, J. H. (1992).** Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. MIT Press. DOI: 10.7551/mitpress/1090.001.0001.
7. **Dorigo, M., Gambardella, L. C., Birattari, M., Martinoli, A., Poli, R., Stützle, T. (2006).** Ant colony optimization and swarm intelligence. 5th international workshop, ANTS 2006, Brussels, Springer.
8. **Gutjahr, W. J. (2002).** Aco algorithms with guaranteed convergence to the optimal solution. Information Processing Letters, Vol. 82, No. 3, pp. 145–153. DOI: 10.1016/s0020-0190(01)00258-7.
9. **Sheikhahmadi, A., Zareie, A. (2020).** Identifying influential spreaders using multi-objective artificial bee colony optimization. Applied Soft Computing, Vol. 94, pp. 106436. DOI: 10.1016/j.asoc.2020.106436.
10. **Khennak, I., Drias, H., Drias, Y., Bendakir, F., Hamdi, S. (2022).** I/f-race tuned firefly algorithm and particle swarm optimization for k-medoids-based clustering. Evolutionary Intelligence, Vol. 16, No. 1, pp. 351–373. DOI: 10.1007/s12065-022-00794-z.
11. **Yang, X. (2013).** Cuckoo search and firefly algorithm: overview and analysis. Studies in Computational Intelligence, Vol. 516, pp. 1–26. DOI: 10.1007/978-3-319-02141-6_1.
12. **Jacobson, S. H., Yücesan, E. (2004).** Analyzing the performance of generalized hill climbing algorithms. Journal of Heuristics, Vol. 10, No. 4, pp. 387–405. DOI: 10.1023/b:heur.0000034712.48917.a9.
13. **Storn, R., Price, K. (1997).** Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, Vol. 11, No. 4, pp. 341–359. DOI: 10.1023/a:1008202821328.
14. **Cordes, K., Rosenhahn, B., Ostermann, J. (2011).** Increasing the accuracy of feature evaluation benchmarks using differential evolution. IEEE Symposium on Differential Evolution, pp. 1–8 DOI: 10.1109/sde.2011.5952056.
15. **Nicolau, M., Krawiec, K., Heywood, M. I., Castelli, M., García-Sánchez, P., Merelo, J. J., Rivas-Santos, V. M., Sim, K. (2014).** Genetic Programming. Lecture Notes in Computer Science, 17th European Conference, EuroGP 2014, Granada, Spain.
16. **Saif, F. A., Latip, R., Hanapi, Z. M., Shafinah, K. (2023).** Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing. IEEE Access, Vol. 11, pp. 20635–20646. DOI: 10.1109/access.2023.3241240.

17. **Saophan, P., Pannakkong, W., Singhaphandu, R., Huynh, V. (2023).** Rapid production rescheduling for flow shop under machine failure disturbance using hybrid perturbation population genetic algorithm-artificial neural networks (PPGA-ANNS). IEEE Access, Vol. 11, pp. 75794–75817. DOI: 10.1109/access.2023.3294573.
18. **Bergeijk, W. A. V. (1961).** Studies with artificial neurons, II: analog of the external spiral innervation of the cochlea. Kybernetik, Vol. 1, No. 3, pp. 102–107. DOI: 10.1007/bf00 290180.
19. **Harmon, L. D. (1961).** Studies with artificial neurons, I: properties and functions of an artificial neuron. Kybernetik, Vol. 1, No. 3, pp. 89–101. DOI: 10.1007/bf00290179.
20. **Levinson, J., Harmon, L. D. (1961).** Studies with artificial neurons, III: Mechanisms of flicker-fusion. Kybernetik, Vol. 1, No. 3, pp. 107–117. DOI: 10.1007/bf00290181.
21. **Kennedy, J., Eberhart, R. (1995).** Particle swarm optimization. Proceedings of the International Conference on Neural Networks, Vol. 4, pp. 1942-1948. DOI: 10.1109/ICNN. 1995.488968.
22. **Sengupta, S., Basak, S., Peters, R. (2018).** Particle swarm optimization: a survey of historical and recent developments with hybridization perspectives. Machine Learning and Knowledge Extraction, Vol. 1, No. 1, pp. 157–191. DOI: 10.3390/make1010010.
23. **Wolpert, D., Macready, W. (1997).** No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1, pp. 67–82. DOI: 10.1109/4235. 585893.
24. **Baig, A. R., Rashid, M. (2007).** Honey bee foraging algorithm for multimodal and dynamic optimization problems. Proceedings of the 9th annual conference on Genetic and evolutionary computation pp. 169. DOI: 10.11 45/1276958.1276983.
25. **Pierezan, J., Coelho, L. D. S. (2018).** Coyote optimization algorithm: A new metaheuristic for global optimization problems. IEEE Congress on Evolutionary Computation, pp. 1-8. DOI: 10.1109/CEC.2018.8477769.
26. **Mohammadi-Balani, A., Nayeri, M. D., Azar, A., Taghizadeh-Yazdi, M. (2021).** Golden eagle optimizer: a nature-inspired metaheuristic algorithm. Computers and Industrial Engineering, Vol. 152, pp. 107050. DOI: 10.1016/j.cie.2020.107050.
27. **Wikelski, M., Moskowitz, D., Adelman, J. S., Cochran, J., Wilcove, D. S., May, M. L. (2006).** Simple rules guide dragonfly migration. Biology Letters, Vol. 2, No. 3, pp. 325–329. DOI: 10.1098/rsbl.2006.0487.
28. **Zadeh, L. (1965).** Fuzzy sets. Information and Control, Vol. 8, No. 3, pp. 338–353. DOI: 10.1016/s0019-9958(65)90241-x.
29. **Kosko, B. (1992).** Neural networks and fuzzy systems: A dynamical systems approach to machine intelligence. Prentice-Hall.
30. **Klir, G. J., Yuan, B. (1995).** Fuzzy sets and fuzzy logic: Theory and applications. Prentice Hall P T R.
31. **Roger-Jang, J. S., Sun, C. T., Mizutani, E. (1997).** Neuro-fuzzy and soft computing: A computational approach to learning and machine intelligence. Prentice Hall.
32. **Yager, R. R., Filev, D. P. (1994).** Essentials of fuzzy modeling and control. 1st Edition, Wiley.
33. **Dubois, D., Prade, H. (1980).** Fuzzy sets and systems: Theory and applications. Academic Press, Vol. 144, pp. 1–393.
34. **Amador-Angulo, L., Ochoa, P., Peraza, C., Castillo, O. (2023).** Fuzzy dynamic adaptation of an artificial fish swarm algorithm for the optimization of benchmark functions. Studies in Computational Intelligence, pp. 99–114. DOI: 10.1007/978-3-031-28999-6_6.
35. **Castillo, O., Lizárraga, E., Soria, J., Melin, P., Valdez, F. (2015).** New approach using ant colony optimization with ant set partition for fuzzy control design applied to the ball and beam system. Information Sciences, Vol. 294, pp. 203–215. DOI: 10.1016/j.ins.2014.09.040.
36. **Amador-Angulo, L., Mendoza, O., Castro, J., Rodríguez-Díaz, A., Melin, P., Castillo, O. (2016).** Fuzzy sets in dynamic adaptation of

parameters of a bee colony optimization for controlling the trajectory of an autonomous mobile robot. Sensors, Vol. 16, No. 9, pp. 1458. DOI: 10.3390/s16091458.

37. **Valdez, F., Vazquez, J. C., Melin, P., Castillo, O. (2017).** Comparative study of the use of fuzzy logic in improving particle swarm optimization variants for mathematical functions using co-evolution. Applied Soft Computing, Vol. 52, pp. 1070–1083. DOI: 10.1016/j.asoc.2016.09.024.
38. **Sánchez, D., Melin, P., Castillo, O. (2017).** A grey wolf optimizer for modular granular neural networks for human recognition. Computational Intelligence and Neuroscience, Vol. 2017, pp. 1–26. DOI: 10.1155/2017/4180510.
39. **González, B., Valdez, F., Melin, P., Prado-Arechiga, G. (2015).** Fuzzy logic in the gravitational search algorithm for the optimization of modular neural networks in pattern recognition. Expert Systems with Applications, Vol. 42, No. 14, pp. 5839–5847. DOI: 10.1016/j.eswa.2015.03.034.
40. **Tai, K., El-Sayed, A., Biglarbegian, M., Gonzalez, C., Castillo, O., Mahmud, S. (2016).** Review of recent type-2 fuzzy controller applications. Algorithms, Vol. 9, No. 2, pp. 39. DOI: 10.3390/a9020039.
41. **Ontiveros, E., Melin, P., Castillo, O. (2020).** Comparative study of interval type-2 and general type-2 fuzzy systems in medical diagnosis. Information Sciences, Vol. 525, pp. 37–53. DOI: 10.1016/j.ins.2020.03.059.
42. **Melin, P., Castillo, O. (2004).** A new method for adaptive control of non-linear plants using type-2 fuzzy logic and neural networks. International Journal of General Systems, Vol. 33, No. 2-3, pp. 289–304. DOI: 10.1080/03081070310001633608.
43. **Moreno, J. E., Sanchez, M. A., Mendoza, O., Rodríguez-Díaz, A., Castillo, O., Melin, P., Castro, J. R. (2020).** Design of an interval type-2 fuzzy model with justifiable uncertainty. Information Sciences, Vol. 513, pp. 206–221. DOI: 10.1016/j.ins.2019.10.042.
44. **Guerrero, M., Valdez, F., Castillo, O. (2022).** Comparative study between type-1 and interval type-2 fuzzy systems in parameter adaptation for the cuckoo search algorithm. Symmetry, Vol. 14, No. 11, pp. 2289. DOI: 10.3390/sym14112289.
45. **Cuevas, F., Castillo, O., Cortés-Antonio, P. (2022).** Generalized type-2 fuzzy parameter adaptation in the marine predator algorithm for fuzzy controller parameterization in mobile robots. Symmetry, Vol. 14, No. 5, pp. 859. DOI: 10.3390/sym14050859.

Article received on 02/02/2024; accepted on 23/04/2024.
**Corresponding author is Fevrier Valdez.*

ISSN 2007-9737

Advancing Cloud Task Scheduling: Recent Developments and Comparative Insights

Jessica González-San-Martín[1,*], Laura Cruz-Reyes[1], Bernabé Dorronsoro[2], Héctor Fraire-Huacuja[1], Marcela Quiroz-Castellanos[3], Claudia Gómez-Santillán[1], Nelson Rangel-Valdez[4]

[1] Tecnológico Nacional de México,
Instituto Tecnológico de Ciudad Madero,
Division of Graduate Studies and Research,
Mexico

[2] University of Cadiz,
Computer Science Engineering,
Spain

[3] Universidad Veracruzana,
Artificial Intelligence Research Center,
Mexico

[4] Tecnológico Nacional de México,
Instituto Tecnológico de Ciudad Madero,
Research Fellow at Graduate Program Division,
Mexico

jessica.gs@cdmadero.tecnm.mx

Abstract. In the present landscape of cloud computing, the effective scheduling of tasks stands as a pivotal element in optimizing the operational efficiency of distributed systems. This paper conducts a thorough and comparative examination of recent trends and progress within this vital and ever-evolving domain. By meticulously reviewing crucial performance metrics and critically analyzing state-of-the-art methodologies, we present a comprehensive overview of Cloud Task Scheduling. We emphasize the shift towards multi-objective strategies, mirroring the escalating complexity and diversity witnessed in cloud environments. Employing innovative approaches and illustrative case studies, we delve into the practical implementation of prominent algorithms, including L_{ABC}, MaOEA-SIN, and MALO. The detailed analysis not only underscores their efficacy in real-world contexts but also pinpoints areas ripe for enhancement and adaptation within multi-cloud settings. Beyond offering an in-depth understanding of the latest developments in Cloud Task Scheduling, this article endeavors to stimulate collaboration and discourse within the academic and professional community. We aim to ignite future advancements, thereby contributing to the sustained growth of this strategic and dynamic field.

Keywords. Cloud task scheduling, cloud computing, strategies and techniques, multi-objective metaheuristics.

1 Introduction

Cloud computing is an information technology service delivery model that allows access to computing resources over the Internet. Instead of owning and maintaining servers and other infrastructure components locally, organizations and users can rent or use cloud services provided by specialized providers. In the current era of cloud computing, Cloud Task Scheduling emerges as an essential component to optimize resource management in distributed systems.

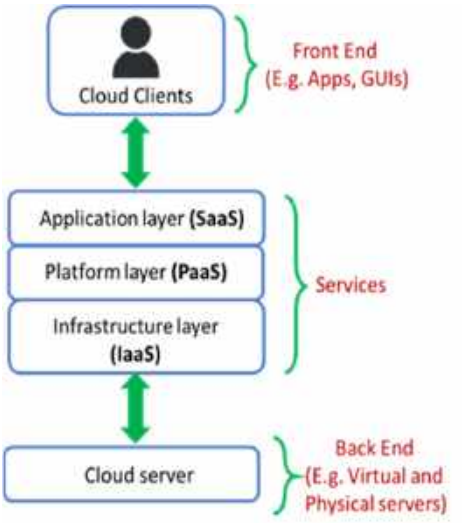


Fig. 1. Diagram by layers of cloud computing architecture [1]

Cloud computing has revolutionized the way organizations manage and access computing resources, offering flexibility and scalability. In this context, efficient task allocation becomes a crucial factor in optimizing performance and resource utilization in cloud environments.

Cloud Task Scheduling refers to the efficient planning and execution of tasks in distributed systems and cloud computing environments.

This discipline seeks to address complex challenges, such as optimal resource allocation, minimization of execution time, and maximization of the utilization of available resources. Given the increasing diversity and complexity of cloud environments, it becomes imperative to adopt advanced approaches to task scheduling.

Instead of relying on single performance metrics, the current trend is toward multi-objective strategies that consider multiple metrics simultaneously. This addresses the need to address multiple objectives and challenges inherent to cloud task scheduling, such as makespan minimization, cost optimization, and load balancing.

The adoption of multi-objective algorithms allows greater flexibility and adaptability to dynamic and heterogeneous cloud environments. This article dives into a detailed exploration of the latest trends and advancements in Cloud Task Scheduling.

Through a review of key performance metrics and critical analysis of cutting-edge methodologies, a comprehensive overview of this vital discipline is provided. The importance of multi-objective approaches is highlighted, and representative case studies will be explored.

The detailed analysis not only highlights effectiveness in real-world scenarios but also identifies areas ripe for improvements and adaptations in multi-cloud environments.

Through this work, we seek to provide an in-depth understanding of the latest developments in Cloud Task Scheduling, as well as foster collaboration within the academic and professional community and highlight future advances to contribute to the continued growth of this research field.

2 Architectural Components of Cloud Computing

The cloud computing architecture is a service model that provides on-demand access to shared computing resources over the Internet. This architecture applied to the Cloud Task Scheduling problem is generally divided into several service models and layers, each with its characteristics and functions. This allows efficient execution of distributed applications. As service models:

– Infrastructure as a Service (IaaS): Offers virtualized computing, storage, and network resources over the Internet. Users can manage and control these resources according to their needs.

- Platform as a Service (PaaS): Provides complete development and execution environments, including database services, middleware, and development tools. Users focus on application development without worrying about managing the underlying infrastructure.
- Software as a Service (SaaS): Offers complete applications through the cloud, generally accessible through a web browser. Users can use these applications without worrying about infrastructure management, updates, or maintenance.

As deployment models:

Algorithm 1 General framework of L_{ABC} algorithm

Input: System parameters

Output: Best solution

```
for i = 1 to P_size do
    Initialize the solution in a random way.
    Evaluate it and insert it into the initial population.
Record the best solution found so far.
while the stopping criterion is not satisfied do
    Employed bee phase
    for i = 1 P_size do
        Set the i^th employed bee on the i^th food source in the current population and perform the explotation task.
        Evaluate the newly generated solution and initialize the adaptive neighborhood structure.
    Onlooker bee phase
    for i = 1 P_size do
        Randomly select three solutions in the current population, select the best one as the food source for the onlooker by using the tournament selection method.
        Perform the exploitation task around the selected food source.
        Evaluate the newly generated solution and update the adaptive neighborhood structure.
        Perform the deep exploitation around the newly generated food source found by the above step.
    Scout bee phase
    If a solution in the population has not been improved during the limit trials, abandon it.
    Generate eight neighboring solutions by using the perturbation structures, and select the best neighboring solution as the scout bee to replace the current solution.
    Deep-exploitation phase
    Perform the deep-exploitation process around the best food source found so far.
    Replace the worst food source in the current population with the best one.
Output Best solution
```

- Public Cloud: Provides services over the Internet for the public. Resources are shared between multiple users and organizations.

- Private Cloud: Resources are used exclusively by one organization. It may be managed internally by the organization or by a specialized service provider. It provides greater control and customization but also involves higher costs.

- Hybrid Cloud: Combines resources from public and private clouds, allowing the portability of data and applications between them. It offers flexibility and the ability to take advantage of the benefits of both implementations.

Finally, within the context of layers, Wei [1] presents the following structure (Fig. 1) that composes the architecture of cloud computing:

– Client Layer (Client): This is the outermost layer and represents the interfaces through which users interact with cloud services. It may include graphical user interfaces, command line interfaces, mobile applications, and other means through which users access and manage cloud resources.

– Application Layer: In this layer are the applications and services that users deploy in the cloud. It may include web applications, business applications, and data analysis services, among others. These applications run on top of the infrastructure provided by the lower layers.

– Platform Layer: The platform layer provides execution environments and services that facilitate the development, deployment, and management of applications. Here is the platform as a service (PaaS), which includes managed databases, application servers, development environments, and other services that allow developers to focus on application logic without worrying about the underlying infrastructure.

– Infrastructure Layer: In this layer, infrastructure as a service (IaaS) is provided that includes computing resources, storage, and networks. Users can provision and manage virtual machines, virtual disks, virtual networks, and other resources as needed.

This layer serves as the base upon which the upper layers are built.

– Cloud Servers: At the lowest level, there are the physical and virtual servers that form the cloud infrastructure.

Algorithm 2 Framework of MaOEA-SIN

Input: The population p, the reference point z
Output: Population P

$O_{min} = \min(p_i)$ *//the minimum value for the objective function*
while $(t < t_{\max})$ **do**
  $d_i = \sin d(p_i)$
  Select two individuals randomly x_1 and x_2
  If $x_1 < x_2$
    $\text{Mating Pool}[P] = \text{Mating Pool}[P] \cup x_1$
  else if $d_{x1} < d_{x2}$
    $\text{Mating Pool}[P] = \text{Mating Pool}[P] \cup x_1$
  $q = \text{crossovermutation}(\text{Mating Pool[p]})$
  $R = [p, q]\ E = ||R_i, O_{min}||$
  for $S = 1 : N$ **do** *//population with size N*
    Select two individuals with minimum angle R_i, R_j
    If $E(R_i) > E(R_j)$
      $R_i \rightarrow []$ *//Eliminating the individual*

Algorithm 3 Ant Lion optimizer algorithm

Initialize the random solutions
Calculate the fitness function
Find the best antlions and assume it as the optimal so far
while the termination criterion is not reached **do**
  for each solution (ant)
  do
    Select an antlion using Roulette wheel
    Update the perimeters e and d
    Create a random walk normalize the chosen random walk
    If $n_i = CP$ **then**
      Update the current solution by using Eq. (15)
    else if $n = CP$ **then**
      Update the current solution by using Eq. (23)
    If $n_i = CP$ **then**
      Update the current solution by using Eq. (23)
    Calculate the fitness function of all solutions using Eq. (13)
    Replace an antlion (new solution) with its corresponding ant (current) if becomes fitter.
  Update the current best solution if an antlion becomes fitter than the old best.
return Thebestsolution(elite)

These servers are managed by cloud service providers and provide the resources necessary to host applications and services. They can include globally distributed data centers to ensure availability and redundancy.

Users and developers mainly interact with the upper layers (Client, Application, and Platform), while cloud service providers manage the underlying infrastructure (Cloud Infrastructure and Servers).

This hierarchical approach facilitates the management and scalability of cloud services, allowing users to focus on application development and deployment without worrying about managing physical infrastructure.

3 Unique Challenges and Opportunities in Cloud Task Scheduling

The dynamic and distributed environment of cloud computing poses several challenges and opportunities when it comes to efficient task scheduling. Task scheduling policies and schemes have direct impacts on effective resource utilization and user task efficiency in the cloud.

Consequently, achieving optimal scheduling and allocation of user tasks remains a very important issue in the field of cloud computing [2].

Below, we will explore some of the most important aspects that professionals in the field propose to address to optimize the performance and effectiveness of task scheduling in this innovative environment. Among the challenges identified are:

- **Variability in Resources:** The shared nature of cloud resources introduces variability in virtual machine performance and resource availability. Task scheduling must be able to adapt to these fluctuations to ensure efficient execution.
- **Network Latency:** The geographic distribution of data centers and reliance on cloud services can lead to significant network latencies. Minimizing the impact of latency on task scheduling becomes a critical challenge.
- **Dynamic Elasticity:** The ability to scale resources on demand is essential in the cloud.

Table 1. Most commonly used performance metrics in cloud task scheduling

Performance metric	Definition
Makespan	The total time from start to completion of all scheduled tasks.
Throughput	The number of tasks completed per unit of time.
Latency	The time a task takes from request to completion.
Resource utilization	The proportion of resources (CPU, memory, etc.) used during task execution.
Resource allocation	The system's ability to allocate resources in an equitable and optimized manner.
Cost	The total expenditure associated with the execution of tasks, considering factors such as the cost of infrastructure and energy.
Load balancing	The equitable distribution of the workload among available resources.
Energy efficiency	The system's ability to perform tasks with the lowest possible energy consumption.

However, effectively implementing dynamic elasticity without compromising performance presents specific challenges.

- **Coordination and Communication:** Effective coordination between distributed tasks and efficient management of communication between components are crucial aspects to avoid bottlenecks and ensure smooth execution of tasks. In the area of opportunities, the following standout:
- **Resource Optimization:** Flexibility in cloud resource allocation provides opportunities to optimize resource utilization, reducing costs and improving energy efficiency.
- **Orchestration Services:** The increasing availability of orchestration services, such as Kubernetes, offers opportunities to simplify the management and coordination of distributed tasks.
- **Predictive Analysis:** Predictive analytics based on historical data can be used to anticipate load patterns and improve decision-making in task scheduling, thereby optimizing performance.
- **Intelligent Automation:** Applying intelligent automation techniques, such as machine learning, can improve adaptive capacity and real-time decision-making to address dynamic cloud challenges.

Exploring these challenges and opportunities will provide a more complete view of the critical aspects to consider when designing effective task scheduling strategies in the cloud computing environment.

4 Innovative Strategies and Techniques

Research in cloud task scheduling has experienced notable advances in the last decade, highlighting innovative strategies and advanced techniques. From 2019 to the present, numerous studies have explored and refined approaches to optimize task allocation in cloud computing environments, creatively addressing changing challenges. In this section, we present a summary of the most recent works in the literature, covering the period from 2019 to the present, and highlighting the strategies used by each.

With a total of 26 studies selected, an important shift towards multi-objective strategies instead of a single objective is highlighted. This change reflects the complexity of cloud environments, where optimizing a single objective may not be enough. Considering multiple performance metrics becomes essential for more adaptable and efficient solutions.

Next, we present works that adopt this multi-objective approach, highlighting the importance of considering multiple performance criteria in cloud task scheduling.

We seek to offer a comprehensive view of the latest innovations, emphasizing the effectiveness of multi-objective approaches in this context.

Pang, 2019 [3]

- Algorithm: EDA-CG
- Year: 2019
- Strategy/Technique: Estimation of distribution algorithm (EDA) and genetic algorithm (GA).
- No. Objectives: 2
- Objectives: Makespan and load balancing.

Langhnoja and Joshiyara, 2019 [4]

- Algorithm: Multi-objective based Integrated Task scheduling.
- Year: 2019
- Strategy/Technique: A ranking method to find the best possible solution.
- No. Objectives: 3
- Objectives: Makespan, cost, and load balancing.

Abdullahi et al., 2019 [5]

- Algorithm: CMSOS
- Year: 2019
- Strategy/Technique: Chaotic optimization strategy and chaotic local search strategy are applied to Pareto Fronts.
- No. Objectives: 2
- Objectives: Makespan and cost.

Abdullah et al., 2019 [6]

- Algorithm: MOPSO and MOPSO_SI
- Year: 2019
- Strategy/Technique: Multi-Objectives PSO (MOPSO) and MOPSO with Importance Strategy (IS).
- No. Objectives: 3
- Objectives: Makespan, cost, and load balancing.

Li and Han, 2020 [7]

- Algorithm: L_{ABC}
- Year: 2020
- Strategy/Technique: Hybrid discrete artificial bee colony (ABC) algorithm and permutation-based encoding method.
- No. Objectives: 3
- Objectives: Makespan, device workload, and total workloads.

Cai et al., 2020 [8]

- Algorithm: MaOEA-SIN
- Year: 2020
- Strategy/Technique: Many-objective intelligent algorithm with sine function.
- No. Objectives: 6
- Objectives: Makespan, cost, throughput, energy, resource utilization, and balancing load.

Singh et al., 2020 [9]

- Algorithm: CPO-MTS
- Year: 2020
- Strategy/Technique: Crow Search optimization Algorithm (CSA) and the Penguin Search Optimization Algorithm (PeSOA).
- No. Objectives: 4
- Objectives: Load balancing, resource utilization, makespan, and Quality of Service.

Abualigah and Diabat, 2021 [10]

- Algorithm: MALO
- Year: 2021
- Strategy/Technique: Hybrid antlion optimization algorithm with elite-based differential evolution.
- No. Objectives: 3
- Objectives: Makespan, response time (CPU), and resource utilization.

Guo, 2021 [11]

- Algorithm: Fuzzy self-defense algorithm
- Year: 2021
- Strategy/Technique: Fuzzy self-defense algorithm.
- No. Objectives: 3
- Objectives: Makespan, load balancing, and cost.

Emara et al., 2021 [12]

- Algorithm: G-MOTSA
- Year: 2021
- Strategy/Technique: Modified genetic algorithm (GA).
- No. Objectives: 6
- Objectives: Makespan, throughput, scheduling length, resource utilization, energy, and imbalance degree.

Kruekaew and Kimpan, 2022 [13]

- Algorithm: MOABCQ
- Year: 2022
- Strategy/Technique: Hybrid artificial bee colony algorithm with reinforcement learning.
- No. Objectives: 3
- Objectives: Makespan, cost, and resource utilization.

Mahmoud et al., 2022 [14]

- Algorithm: TS-DT
- Year: 2022
- Strategy/Technique: Multi-objective task scheduling algorithm is proposed based on the decision tree.
- No. Objectives: 3
- Objectives: Makespan, resource utilization, and load balancing.

Mangalampalli et al., 2022 [15]

- Algorithm: CSO
- Year: 2022
- Strategy/Technique: Cat Swarm Optimization algorithm.
- No. Objectives: 4
- Objectives: Makespan, migration time, energy, and cost.

Mangalampalli et al., 2023 [16]

- Algorithm: MOTSGWO
- Year: 2023
- Strategy/Technique: Grey wolf optimization algorithm.
- No. Objectives: 3
- Objectives: Makespan, migration time, and energy.

Cui et al., 2023 [17]

- Algorithm: MO-MFO
- Year: 2023
- Strategy/Technique: Evolutionary multi-factorial optimization algorithm.
- No. Objectives: 3
- Objectives: Makespan, cost, and load balancing.

Chandrashekar et al., 2023 [18]

- Algorithm: HWACO
- Year: 2023
- Strategy/Technique: Hybrid Weighted Ant Colony Optimization algorithm.
- No. Objectives: 2
- Objectives: Makespan and cost.

Agarwal et al., 2023 [19]

- Algorithm: HGA-ECS
- Year: 2023
- Strategy/Technique: Integration of Genetic Algorithm (GA) and Energy Conscious Scheduling (ECS) model.
- No. Objectives: 3
- Objectives: Makespan, energy consumption, and optimization of task scheduling over processors.

Mangalampalli et al., 2023 [20]

- Algorithm: MOTSWAO
- Year: 2023
- Strategy/Technique: Whale Optimization Algorithm.
- No. Objectives: 2
- Objectives: Makespan and energy consumption.

Malti et al., 2023 [21]

- Algorithm: Hybrid Multi-objective Optimization Algorithm.
- Year: 2023

- Strategy/Technique: Combination of flower pollination behavior and grey wolf optimizer strategy for task scheduling optimization.
- No. Objectives: 4
- Objectives: Makespan, resource utilization, degree of imbalance, and maximization of throughput in heterogeneous IaaS cloud environments.

Pirozmand et al., 2023 [22]

- Algorithm: IPSO
- Year: 2023
- Strategy/Technique: Multi-adaptive learning strategy to shorten the execution time of the original PSO algorithm.
- No. Objectives: 3
- Objectives: Makespan, load balancing and execution time.

Khan, 2024 [23]

- Algorithm: HLFO
- Year: 2024
- Strategy/Technique: Convolutional and Recurrent Neural Networks in a deep learning model for load calculation, Reinforcement Learning with a Hybrid Lyrebird Falcon Optimization (HLFO) algorithm.
- No. Objectives: 4
- Objectives: Makespan, energy consumption, resource utilization and Quality of Service (QoS).

Sabat et al., 2024 [24]

- Algorithm: Adaptive PSO-ACO
- Year: 2024
- Strategy/Technique: Adaptive particle swarm optimization (PSO) and ant colony optimization (ACO).
- No. Objectives: 3
- Objectives: Cost, makespan and execution time.

Gupta and Singh, 2024 [25]

- Algorithm: WOA-Scheduler
- Year: 2024
- Strategy/Technique: Whale Optimization Algorithm.
- No. Objectives: 3
- Objectives: Cost, makespan and load balancing.

Ciptaningtyas et al., 2024 [26]

- Algorithm: Improved Squirrel Search Algorithm (SSA)
- Year: 2024
- Strategy/Technique: Integration with Opposition Based Learning (OBL) method to address premature convergence.
- No. Objectives: 3
- Objectives: Makespan, throughput, and resource utilization.

Nithiavathy et al., 2024 [27]

- Algorithm: AGDESMA
- Year: 2024
- Strategy/Technique: Slime Mould Algorithm (SMA) and Adaptive Guided Differential Evolution (AGDE).
- No. Objectives: 2
- Objectives: Makespan and cost.

Behera and Sobhanayak, 2024 [28]

- Algorithm: Hybrid GA-GWO
- Year: 2024
- Strategy/Technique: Grey Wolf Optimization Algorithm (GWO) and the Genetic Algorithm (GA).
- No. Objectives: 3
- Objectives: Makespan, cost and energy consumption.

5 Critical Examination of State-of-the-Art Methods

The previous section has provided an overview of the most recent works in cloud task scheduling, highlighting innovative strategies and advanced techniques used by various researchers. Now, we will delve into a critical analysis focused on the three most representative works in this collection.

Table 2. Results obtained in [7] for the different objectives that L_{ABC}addresses

Problem	L_{ABC}					
	f_1	f_2	f_3	Fitness Value	Average Makespan	Time (s)
1	23+	19	119.65	41.13+	23+	1.25
2	297+	193.6	1642.7	535.12+	297+	0.53

Table 3. Results comparison presented in [7] of L_{ABC} algorithm against other approaches in the literature in terms of makespan

Problem	L_{ABC}		AIS		SFLA		EDA	
	f_1	Average Makespan	f_1	Average	f_1	Average.	f_1	Average
1	23+	23+	27	27	24	24	23	23.4
2	297+	297+	-	-	297	307.3	297	297.4

Table 4. Parameter settings for cloud simulation [8]

Cloud	Mips	Cost	Bandwidth	Transmission
Cloud1	300-450	0.03	1024-2048	0.01
Cloud2	500-1000	0.06	2048-3072	0.02
Cloud3	1500-2000	0.09	3072-4096	0.03

Each of these studies has contributed to the evolution of methods and approaches in task allocation optimization. We will break down in detail the techniques used, and the algorithms implemented by these selected works.

By critically examining these notable studies, we seek to provide an in-depth understanding of the key contributions that have driven the current state of the art in cloud task scheduling. This analysis will not only illustrate the strengths and limitations of each approach but will also establish a solid framework for understanding the broader landscape of research in this dynamic and constantly evolving field.

5.1 Hybrid Multi-Objective Artificial Bee Colony Algorithm

Li and Han [7] proposed an algorithm called L_{ABC}, a hybrid and improved version of the artificial bee colony (ABC) algorithm. In this approach, the initial problem is modeled as a hybrid flow shop scheduling (HFS) problem, addressing both single and multiple objectives. In the context of multi-objective HFS problems, three objectives are simultaneously considered: minimizing the makespan, the maximum workload on the device, and the total workloads on all devices. The scope of the algorithm extends to two distinct types of HFS: those with identical parallel machines and those involving unrelated machines. The proposed approach incorporates three categories of artificial bees, namely employed, observer, and scout bees, similar to the classical ABC scheme.

Each solution is represented by a string of integers. To adapt to the particularities of the problem, various perturbation structures are explored, and designed to improve the search capabilities of the algorithm.

The inclusion of an improved version of the adaptive perturbation structure in the proposed algorithm stands out, which seeks to effectively balance the exploitation and exploration capacity during the optimization process. A simple but highly effective selection strategy, along with an updated approach, is implemented to enhance the exploitation process.

To further intensify mining capabilities, a deep mining operator is introduced. In addition, an improved version of the scout bee is introduced that uses various local search methods to find the best food source or abandoned solution. This approach significantly contributes to improving the convergence ability of the proposed algorithm.

Table 5. Numerical analysis of different algorithms vs MaOEA-SIN with six objectives [8]

Algorithm	Total time (Min)	Cost (Min)	Throughput (Max)	L. Balancing (Min)	RU (Max)	Energy (Min)
	Average					
NSGA-III	5.2260×10^{5}	9.8234×10^{5}	3.9296×10^{-4}	2.6874×10^{7}	6.7570×10^{-1}	3.5439×10^{4}
VaEA	3.8351×10^{5}	7.7333×10^{5}	3.8251×10^{-4}	2.2864×10^{7}	7.9762×10^{-1}	2.7588×10^{4}
GrEA	4.7793×10^{5}	8.6360×10^{5}	3.7990×10^{-4}	2.6619×10^{7}	7.3459×10^{-1}	3.3846×10^{4}
Two_Arch2	4.7032×10^{5}	9.8155×10^{5}	3.8555×10^{-4}	2.5797×10^{7}	7.3034×10^{-1}	3.3580×10^{4}
KnEA	4.5334×10^{5}	9.0127×10^{5}	**4.0737×10^{-4}**	2.5005×10^{7}	7.3168×10^{-1}	3.2529×10^{4}
MaOEA-SIN	**2.9254×10^{5}**	**5.4047×10^{5}**	4.0393×10^{-4}	**1.9016×10^{7}**	**9.2716×10^{-1}**	**2.1341×10^{4}**

The effectiveness of the algorithm is tested using widely recognized benchmark instance sets, and performance verification of the proposed algorithm is performed.

5.2 Many-Objective Intelligent Algorithm with Sine Function

Cai et al. [8] developed a multi-objective distributed programming model that covers six objectives: total time, cost, cloud performance, energy consumption, resource utilization, and load balancing.

Furthermore, they introduced an intelligent multi-objective algorithm with a sine function to implement this model, called MaOEA-SIN. This algorithm considers the variation trend of the diversity strategy in the population, modeling it in an analogous way to the sine function.

The experimental results show outstanding programming efficiency, which contributes to improving security. This work presents a new perspective to address the challenging problem of data processing in the Internet of Things.

5.3 Multi-Objective Optimization Method using Hybrid Antlion Optimizer Algorithm

Abualigah and Diabat [10] introduced an innovative algorithm, called MALO, that combines antlion optimization with elite-based differential evolution to solve multi-objective task scheduling problems in cloud computing environments. In this method, the multi-objective nature of the problem arises from the need to minimize the makespan and maximize the resource utilization simultaneously.

The antlion optimization algorithm was improved by incorporating elite-based differential evolution as a local search technique. This approach improves the exploitability of the algorithm and prevents the possibility of getting trapped in local optima. The obtained results revealed that MALO outperformed other well-known optimization algorithms.

Notably, MALO showed faster convergence compared to other approaches when applied to larger search spaces, positioning it as a suitable option to address large-scale programming problems. In addition, a statistical analysis was carried out using t-tests, evidencing a significant improvement in the results obtained by MALO. The comprehensive evaluation of leading methods in cloud task scheduling reveals a diversity of innovative approaches and advanced strategies.

The three works examined have proven to be pioneers in the development of efficient and effective solutions to the challenges inherent in this dynamic field. Together, these works have not only contributed significantly to the current state of the art in cloud task scheduling but also provided valuable insights and foundations for future research in this dynamic and challenging field.

6 Performance Metrics and Benchmarks

In the dynamic and challenging realm of Cloud Task Scheduling, accurate evaluation of algorithm performance becomes a crucial component for efficient solution development and deployment. To carry out this evaluation, an essential set of tools is used: performance metrics and benchmarks.

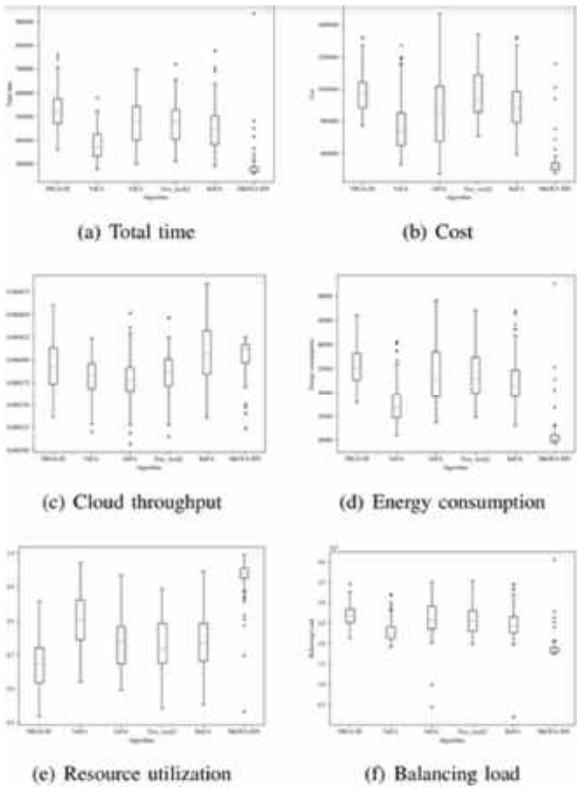


Fig. 2. Comparison of performance presented in [8] of different algorithms on six objectives

Table 6. CloudSim test settings [10]

Element	Parameter	Values
Datacenter	No. of datacenter	2
Cloudlet	No. of cloudlets	100-1000
	Length	1000-2000
Virtual machine	RAM	512 MB
	MIPS	100-1000
	Size	10000
	Bandwidth	1000
	Policy type	Time Shared
	No. of CPUs	1
Host	No. of Hosts	2
	RAM	2048 MB
	Storage	1 million
	Bandwidth	10000

These tools provide the foundation upon which researchers and developers can measure, compare, and continually improve the performance of cloud task scheduling algorithms.

6.1 Performance Metrics

Performance metrics play an essential role in evaluating and improving algorithms. These measures quantify the effectiveness and efficiency of an algorithm by providing objective information about its performance on various tasks.

Using performance metrics, developers and data scientists can evaluate effectiveness, compare algorithms, optimize parameters, diagnose problems, and perform sensitivity analysis, among other actions [29].

In the context of cloud computing, various performance metrics have been used to evaluate the efficiency and effectiveness of the algorithms used, offering detailed insight into system performance, and assisting developers in making informed decisions. Below, Table 1 presents some of the most used performance metrics in Cloud Task Scheduling.

6.2 Benchmarks

Test instances or benchmarks are collections of data created for the specific purpose of evaluating and testing algorithms. The use of instances in the evaluation of algorithms in Cloud Task Scheduling provides a structured and objective framework to analyze and improve the performance of solutions in a dynamic and distributed environment.

These instances provide an accurate representation of real-world challenges and scenarios, allowing developers to make informed decisions and refine their approaches.

There are different sets of instances widely used for the evaluation of Cloud Task Scheduling algorithms. Some notable examples include:

– Google Cluster-Trace Dataset (GoCJ) [30]: GoCJ provides real traces of jobs and tasks executed on Google clusters. It contains valuable insights into the variability and dynamics of work in large-scale cloud environments.

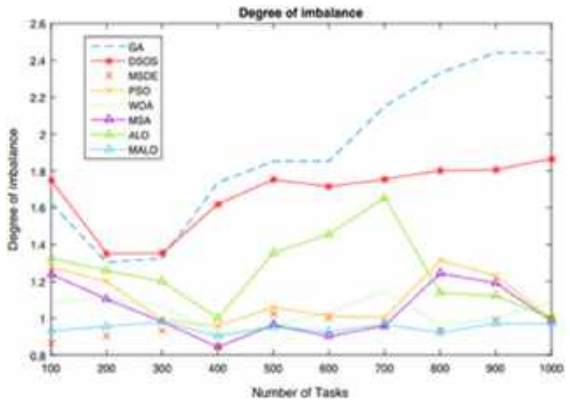


Fig. 3. Degree of imbalance of scheduling algorithms presented in [10]

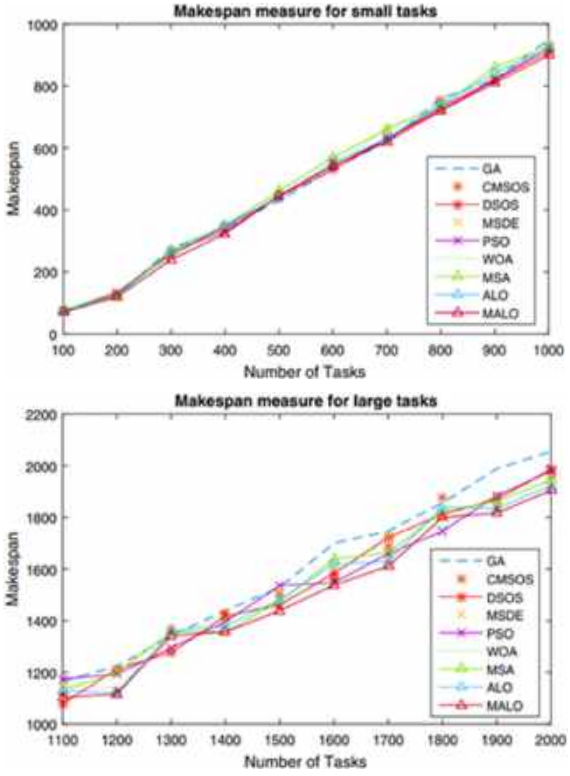


Fig. 4. The average makespan values for executing small and large tasks [10]

– NASA Ames iPSC/860 [31]: This set of instances is based on execution traces of scientific applications on the NASA Ames Research Center iPSC/860 supercomputer. Provides realistic data on scientific workloads in high-performance environments.
– HPC2N-2002 [32]: Derived from execution traces at the High-Performance Computing Center North (HPC2N) in Sweden in 2002. Contains information about the execution of jobs on a high-performance cluster.
– CEC 2005 Benchmark Functions [33]: Although most associated with benchmarking functions for optimization algorithms, the CEC 2005 instance set is also used in some cases to evaluate Cloud Task Scheduling algorithms.

These instance sets are used by the research community to evaluate and compare Cloud Task Scheduling algorithms in various contexts. Each data set presents specific characteristics that allow different aspects of performance in cloud computing environments to be simulated and analyzed.

On the other hand, there is CloudSim [34], which is a simulation framework that provides sets of simulated instances for the evaluation of cloud task scheduling algorithms. It allows you to create simulated cloud environments for performance evaluations. Programmers can generate tasks, virtual machines, and hosts randomly and with different characteristics.

CloudSim is widely used by researchers to evaluate their algorithms in different generated environments.

7 Study Cases and Future Directions

In this section, we present a detailed analysis of case studies that highlight the practical applications and results obtained by the three selected approaches: L_{ABC} [7], MaOEA-SIN [8], and MALO [10]. We will mention the instances used in their experiments, as well as the conditions under which they were carried out and the results obtained.

Through this case study analysis, we seek to provide a deeper understanding of the performance of these algorithms in real-world situations, considering different data sets and application scenarios. Furthermore, we will outline possible future directions that arise from the lessons learned and the results obtained, thus helping to guide subsequent research in task scheduling in cloud computing environments.

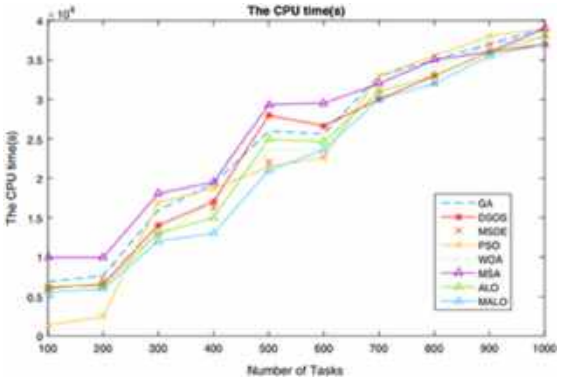


Fig. 5. The CPU time(s) of the task scheduling algorithms for the synthetic datasets [10]

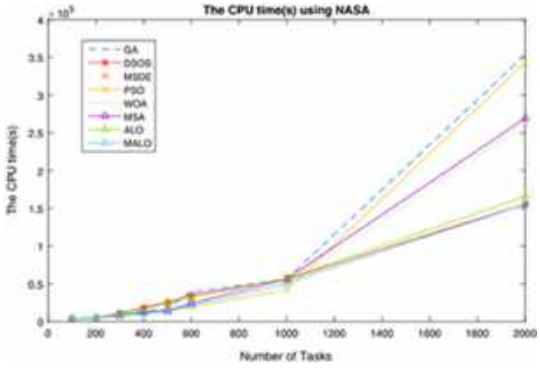


Fig. 7. The CPU time(s) of the task scheduling algorithms for solving the NASA Ames datasets [10]

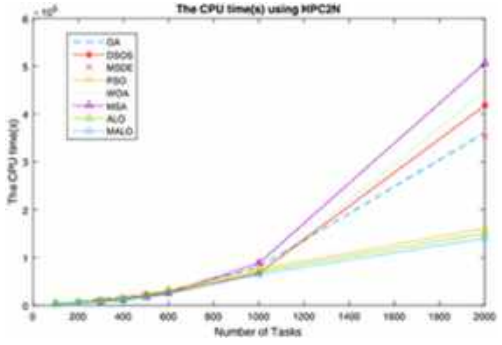


Fig. 6. The CPU time(s) of the task scheduling algorithms for solving the HPC2N Seth datasets [10]

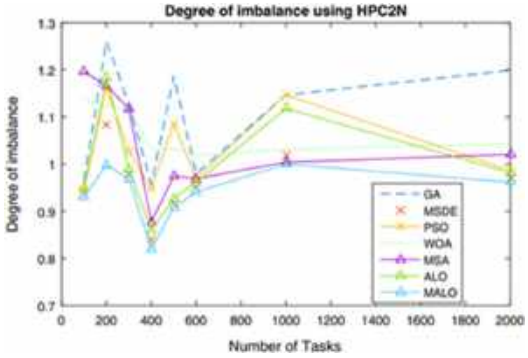


Fig. 8. The degree of imbalance of the tasks scheduling optimization algorithms using the HPC2N Seth datasets [10]

7.1 Study Cases

Li and Han Li and Han compared their proposed L_{ABC} [7] against three different algorithms from the literature: EDA (Wang et al. [35]), AIS (Liu et al. [36]), and SFLA (Xu et al. [37]). For the experimentation, they used a computer with a 3.3 GHz Intel Core i5 processor and 4 GB memory.

To test the performance of the algorithms in a multi-objective environment, they selected as instances two unrelated real machine HFS problems from [35] to make the problem more like reality in a cloud system.

The results obtained by L_{ABC} are presented in Table 2 and 3 revealing the following highlights:

1 When analyzing the comparison of results for each instance, superior performance by the algorithm is evident.

2 In terms of the average value of makespan, the computational results generated by the algorithm match the optimal values for each instance on average, thus underlining the robustness of L_{ABC}.

3 Considering the calculation times used in the test instances, the L_{ABC} algorithm also exhibits superior performance.

The L_{ABC} algorithm according to Li and Han [7] stands out for its competitive performance against various efficient algorithms. This success is based on the introduction of eight meticulously designed disturbance structures to improve exploitation

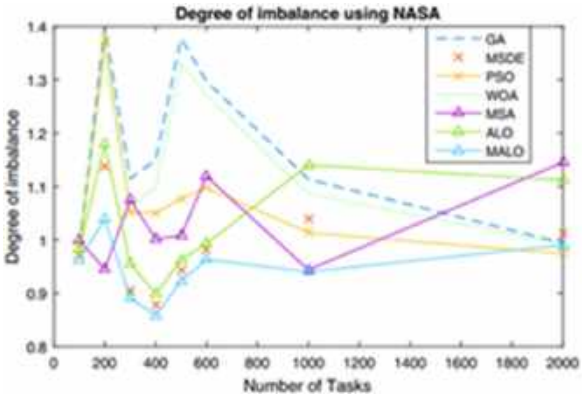


Fig. 9. The degree of imbalance of the tasks scheduling optimization algorithms using the NASA Ames datasets [10]

capacity, the use of an improved deep exploitation observer bee mechanism to intensify local search, the implementation of an adaptive perturbation that balances exploitation and exploration, and a specific approach for scout bees that boosts the convergence capacity of the algorithm.

Cai et al. [8] conducted their experiments through the simulation of three clouds with different characteristics, the main difference is the Mips execution speed and the cost. The execution cost of the first cloud is cheaper, but the execution speed is slower and takes longer, which is suitable for smaller tasks. The second cloud has a relatively medium execution speed and execution cost.

The third cloud has a faster execution speed, but the cost also becomes high, suitable for performing larger tasks. Table 4 presents the detailed parameter settings of the simulated clouds. The experimentation was carried out by generating 300 tasks with an initial length of 500 million instructions (MI), the file size is 200 KB and the output file size is 100 KB. Then each task gradually increases with a trend of 500 MI, 10 KB, and 10 KB respectively.

The comparison of the MaOEA-SIN algorithm was carried out against five algorithms from the literature: NSGA-III, VaEA, GrEA, Two_Arch2, and KnEA. The study evaluated the performance of the MaOEA-SIN algorithm in a multi-cloud model by comparing six objective values.

The best, worst, and average solutions were selected based on the performance of these values. Table 5 presents six objectives: Total time, cost, cloud throughput, load balancing, resource utilization (RU) and Energy consumption, for six different approaches.

The average results presented in Table 5, indicate that MaOEA-SIN outperforms GrEA in cost, although it shows inferior performance in the cloud throughput objective. MaOEA-SIN stands out for its excellent average performance, demonstrating strong convergence.

Figure 2 shows the distribution of the data in the form of a box plot. Looking at the upper quartile, the median, and the lower quartile, the MaOEA-SIN algorithm has better convergence and distribution for six objectives. Convergence is reflected in the fact that the midline values are all optimal values in each objective.

The distribution is reflected in the fact that the MaOEA-SIN algorithm has more dispersed points. In summary, the performance of the MaOEA-SIN algorithm is superior to that of other algorithms. To validate the effectiveness of the MALO algorithm, Abualigah and Diabat [10] present two series of experiments using synthetic datasets and real trace datasets.

For the first set of instances, the Cloudsim environment was used, which is a set of tools to imitate cloud computing scenarios [34], because the investigation of new procedures or approaches in the real cloud computing ecosystem is usually limited by solid foundations, such as protection, security, speed and the high cost of money if experiments are carried out.m Therefore, it is difficult to conduct such research in repeatable, reliable, and scalable ecosystems (environments) using real world cloud environments [10].

For experimentation, they built two data centers within CloudSim, each with two hosts. Each host has 20 GB of RAM (one host is a dual-core machine and the other is a quad-core machine) and one TB of memory storage.

Each host has a collective processing power of one million MIPS. Several virtual machines were designed with different distributions generated such as 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, and 2000 instances. The CloudSim configuration is presented in Table 6. The MALO algorithm is compared against seven approaches from the literature: (Genetic Algorithm (GA) [38], Discrete Symbiotic Organism Search (DSOS) Algorithm [39], Hybrid Moth Search Algorithm

(MSDE) [40], Particle Swarm Optimization (PSO) Algorithm [41], Whale Optimization Algorithm (WOA) [42], Moth Search Algorithm (MSA) [43], and Antlion Optimizer (ALO) Algorithm [44]).

Figure 3 presents the results obtained by the algorithms in terms of the degree of imbalance (DI), this metric shows us how equitably the tasks are distributed in the different resources. A lower degree of imbalance translates to a better use of resources and here we can see that MALO obtains the best DI compared to the other algorithms.

On the other hand, Figure 4 shows the average makespan values obtained by the algorithms for small and large tasks. The MALO algorithm reduced the value of makespan in all task cases. It is concluded that the value of the makespan increases slowly as the size of the tasks increases.

The average value of makespan when using the modified optimization algorithms is better than traditional optimization algorithms [10]. Meanwhile, the average time interval of the MALO algorithm is smaller than that of other comparative methods. Figure 5 displays the response times (CPU) achieved by different task scheduling algorithms (GA, DSOS, MSDE, PSO, WOA, MSA, ALO, and the proposed MALO).

It is highlighted that MALO achieved minimum response times to solve problems of various sizes compared to the other methods, indicating a significant improvement in the efficiency of the algorithm. Specifically, for a task size of 600, the PSO algorithm recorded the lowest response time compared to other methods. For evaluation results of real trace datasets Abualigah and Diabat used the NASA Ames dataset [32] and the HPC2N Seth dataset [33].

Figures 6 and 7 present the response times (CPU) of various task scheduling algorithms (GA, DSOS, MSDE, PSO, WOA, MSA, ALO, and MALO) when performing tasks with real trace data sets. In Figure 6, MALO manages to almost reach the minimum response time for solving tasks of all sizes compared to other methods, especially using the HPC2N Seth datasets.

Similarly, in Figure 7, MALO stands out in approaching the minimum response time for tasks of all sizes, especially with the NASA Ames datasets. The difference in algorithm response times is evident across all task sizes, particularly with the HPC2N Seth datasets, standing out over other methods. Although the difference in response times of the MALO algorithm across all task sizes is not as clear compared to DSOS, an overall improvement is observed that contributes to the reduction in the time needed to find optimal solutions.

The degree of imbalance results between the MALO algorithm and other benchmark algorithms are presented in Figures 8 and 9 for the HPC2N Seth and NASA Ames datasets. MALO achieved a higher load balance (lower degree of imbalance) compared to the other methods. In almost all cases of data sets (100-2000), MALO exhibited the lowest degree of imbalance, highlighting its superior performance compared to comparative optimization algorithms.

This is reflected in a better balance between virtual machines in all problem instances. bMALO converged faster than the other approaches for larger search spaces, making it suitable for large scheduling problems.

7.2 Future Directions

Based on insights gained from case studies of three prominent algorithms in Cloud Task Scheduling [7,8,10], we identify promising directions for future research and development.

1 Improvements in Hybrid Algorithms:

- Inspired by the success of L_{ABC} in introducing innovative perturbation structures, there is potential to explore new hybrid algorithms that combine different optimization strategies for improved performance.
- Investigating advanced exploitation mechanisms and adaptive perturbation strategies, as demonstrated in L_{ABC}, can be crucial to address the constantly evolving challenges in Cloud Task Scheduling.

2 Multi-Cloud Environments:

- Given the dynamic nature of cloud environments, future studies could focus on multi-cloud scenarios with variable execution speeds and costs. This aligns with the approach of Cai et al. by simulating three clouds with different characteristics.

- Exploring adaptive algorithms capable of dynamically adjusting to various cloud configurations can improve the adaptability of scheduling algorithms.

3 Real World Cloud Experiments:

- Bridging the gap between simulations and real-world cloud environments remains a challenge. Future research could explore methodologies for conducting repeatable, reliable, and scalable experiments in real-world cloud ecosystems, considering factors such as security, cost, and efficiency.
- Addressing the limitations associated with real-world experiments would significantly contribute to the practical applicability of the proposed scheduling algorithms.

4 Expansion of Performance Metrics:

- Extending the set of performance metrics beyond traditional objectives, such as exploring energy efficiency, security, and adaptability, can offer a more comprehensive evaluation of scheduling algorithms.
- Investigating the impact of scheduling decisions on the overall sustainability and security of cloud systems would be a valuable avenue for future research.

5 Optimization for Large-Scale Problems:

- The success of MALO in handling larger search spaces suggests the need for algorithms that can scale efficiently for large programming problems.
- Future studies could explore optimization techniques specifically designed to handle the complexity and scale associated with task scheduling in expansive cloud environments.

6 Dynamic Workloads and Task Characteristics:

- Adapting algorithms to accommodate dynamic workloads and diverse task characteristics is crucial. Future research could focus on developing scheduling approaches capable of dynamically adjusting to varying task requirements and environmental conditions.

These future directions aim to guide researchers and practitioners in advancing the field of Cloud Task Scheduling, addressing emerging challenges, and ensuring the continued evolution of efficient and adaptive scheduling algorithms.

8 Conclusions

This comprehensive study explores Cloud Task Scheduling, analyzing recent developments and offering valuable comparative insights. The importance of performance metrics and benchmarks in the evaluation of algorithms is highlighted, underlining their critical role in continuous improvement.

We provide a comprehensive overview of the latest work in the literature, highlighting the richness of multi-objective approaches that seek to simultaneously improve multiple performance metrics. This shift toward more complex and comprehensive strategies reflects the growing awareness of the multifaceted and challenging nature of cloud task scheduling.

Detailed analysis of case studies, including L_{ABC}, MaOEA-SIN and MALO, provide significant insights into their strengths and areas for improvement. The applicability and robustness of these approaches are highlighted in multi-cloud environments and with real data sets.

Promising future directions are identified, from improvements in hybrid algorithms to adaptation to real cloud environments and exploration of additional metrics.

This article aims to contribute to the continued growth of the field by providing an in-depth overview of recent developments. We seek to foster collaboration and dialogue in the academic and professional community, paving the way for future achievements in cloud task scheduling.

References

1. **Wei, X. (2020).** Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing. Journal of Ambient Intelligence and Humanized Computing, pp. 1–12. DOI: 10.1007/s12652-020-02614-7.
2. **Madni, S. H. H., Abd Latiff, M. S., Coulibaly, Y. (2016)**. Resource scheduling for

infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities. Journal of Network and Computer Applications, Vol. 68, pp. 173–200. DOI: 10.10 16/j.jnca.2016.04.016.

3. **Pang, S., Li, W., He, H., Shan, Z., Wang, X. (2019).** An EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing. IEEE Access, Vol. 7, pp. 146379–146389. DOI: 10.1109/access.2019.2946216.
4. **Langhnoja, H. K., Joshiyara, P. H. A. (2019).** Multi-objective based integrated task scheduling in cloud computing. 3rd International conference on Electronics, Communication and Aerospace Technology. DOI: 10.1109/iceca.2019.8821912.
5. **Abdullahi, M., Ngadi, M. A., Dishing, S. I., Abdulhamid, S. M., Ahmad, B. I. (2019).** An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment. Journal of Network and Computer Applications, Vol. 133, pp. 60–74. DOI: 10.1016/j.jnca.2019.02.005.
6. **Abdullah, M., Al-Muta'a, E. A., Al-Sanabani, M. (2019)**. Integrated MOPSO algorithms for task scheduling in cloud computing. Journal of Intelligent & Fuzzy Systems, Vol. 36, No. 2, pp. 1823–1836. DOI: 10.3233/jifs-181005.
7. **Li, J., Han, Y. (2019).** A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system. Cluster Computing, Vol. 23, No. 4, pp. 2483–2499. DOI: 10.1007/s10586-019-03022- z.
8. **Cai, X., Geng, S., Wu, D., Cai, J., Chen, J. (2021).** A multicloud-model-based many-objective intelligent algorithm for efficient task scheduling in internet of things. IEEE Internet of Things Journal, Vol. 8, No. 12, pp. 9645–9653. DOI: 10.1109/jiot.2020.3040019.
9. **Singh, H., Tyagi, S., Kumar, P. (2020).** Crow–penguin optimizer for multiobjective task scheduling strategy in cloud computing. International Journal of Communication Systems, Vol. 33, No. 14, pp. e4467. DOI: 10.1 002/dac.4467.
10. **Abualigah, L., Diabat, A. (2020).** A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. Cluster Computing, Vol. 24, No. 1, pp. 205–223. DOI: 10.1007/ s10586-020-03075-5.
11. **Guo, X. (2021).** Multi-objective task scheduling optimization in cloud computing based on fuzzy self-defense algorithm. Alexandria Engineering Journal, Vol. 60, No. 6, pp. 5603–5609. DOI: 10.1016/j.aej.20 21.04.051.
12. **Emara, F. A., Gad-Elrab, A., Sobhi, A., Raslan, K. R. (2021).** Genetic-based multi-objective task scheduling algorithm in cloud computing environment. International Journal of Intelligent Engineering and Systems, Vol. 14, No. 5, pp. 571–582. DOI: 10.22266/ijies20 21.1031.50.
13. **Kruekaew, B., Kimpan, W. (2022).** Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning. IEEE Access, Vol. 10, pp. 17803–17818. DOI: 10.1109/access.2022. 3149955.
14. **Mahmoud, H., Thabet, M., Khafagy, M. H., Omara, F. A. (2022).** Multiobjective task scheduling in cloud environment using decision tree algorithm. IEEE Access, Vol. 10, pp. 36140–36151. DOI: 10.1109/access.2022. 3163273.
15. **Mangalampalli, S., Swain, S. K., Mangalampalli, V. K. (2021).** Multi objective task scheduling in cloud computing using cat swarm optimization algorithm. Arabian Journal for Science and Engineering, Vol. 47, No. 2, pp. 1821–1830. DOI: 10.1007/s13369-021-06076-7.
16. **Mangalampalli, S., Karri, G. R., Kumar, M. (2022).** Multi objective task scheduling algorithm in cloud computing using grey wolf optimization. Cluster Computing, Vol. 26, No. 6, pp. 3803–3822. DOI: 10.1007/s10586-022-03786-x.
17. **Cui, Z., Zhao, T., Wu, L., Qin, A. K., Li, J. (2023).** Multi-objective cloud task scheduling optimization based on evolutionary multi-factor algorithm. IEEE Transactions on Cloud

Computing, Vol. 11, No. 4, pp. 3685–3699. DOI: 10.1109/tcc.2023.3315014.

18. **Chandrashekar, C., Krishnadoss, P., Poornachary, V. K., Ananthakrishnan, B., Rangasamy, K. (2023).** HWACOA scheduler: Hybrid weighted ant colony optimization algorithm for task scheduling in cloud computing. Applied Sciences, Vol. 13, No. 6, pp. 3433. DOI: 10.3390/app13063433.
19. **Agarwal, G., Gupta, S., Ahuja, R., Rai, A. K. (2023).** Multiprocessor task scheduling using multi-objective hybrid genetic algorithm in fog–cloud computing. Knowledge-Based Systems, Vol. 272, pp. 110563. DOI: 10.1016/j.knosys.2023.110563.
20. **Mangalampalli, S., Karri, G. R., Kose, U. (2023).** Multi objective trust aware task scheduling algorithm in cloud computing using whale optimization. Journal of King Saud University - Computer and Information Sciences, Vol. 35, No. 2, pp. 791–809. DOI: 10.1016/j.jksuci.2023.01.016.
21. **Malti, A. N., Hakem, M., Benmammar, B. (2023).** A new hybrid multi-objective optimization algorithm for task scheduling in cloud systems. Cluster Computing. DOI: 10.1007/s10586-023-04099-3.
22. **Pirozmand, P., Jalalinejad, H., Hosseinabadi, A. A. R., Mirkamali, S., Li, Y. (2023).** An improved particle swarm optimization algorithm for task scheduling in cloud computing. Journal of Ambient Intelligence and Humanized Computing, Vol. 14, No. 4, pp. 4313–4327. DOI: 10.1007/s12652-023-04541-9.
23. **Khan, A. R. (2024).** Dynamic load balancing in cloud computing: optimized RL-based clustering with multi-objective optimized task scheduling. Processes, Vol. 12, No. 3, pp. 519. DOI: 10.3390/pr12030519.
24. **Sabat, N. R., Sahoo, R. R., Pradhan, M. R., Acharya, B. (2024).** Hybrid technique for optimal task scheduling in cloud computing environments. TELKOMNIKA (Telecommunication Computing Electronics and Control), Vol. 22, No. 2, pp. 380. DOI: 10.12928/telkomnika.v22i2.25641.
25. **Gupta, S., Singh, R. S. (2024).** User-defined weight based multi objective task scheduling in cloud using whale optimization algorithm. Simulation Modelling Practice and Theory, Vol. 133, pp. 102915. DOI: 10.1016/j.simpat.2024.102915.
26. **Ciptaningtyas, H. T., Shiddiqi, A. M., Purwitasari, D., Rosyadi, F. D., Fauzan, M. N. (2024).** Multi-objective task scheduling algorithm in cloud computing using improved squirrel search algorithm. International Journal of Intelligent Engineering & Systems, Vol. 17, No. 1.
27. **Nithiavathy, R., Janakiraman, S., Priya, M. D. (2023).** Adaptive guided differential evolution-based slime mould algorithm-based efficient multi-objective task scheduling for cloud computing environments. Transactions on Emerging Telecommunications Technologies, Vol. 35, No. 1. DOI: 10.1002/ett.4902.
28. **Behera, I., Sobhanayak, S. (2024).** Task scheduling optimization in heterogeneous cloud computing environments: a hybrid GA-GWO approach. Journal of Parallel and Distributed Computing, Vol. 183, pp. 104766. DOI: 10.1016/j.jpdc.2023.104766.
29. **González-San-Martín, J., Cruz-Reyes, L., Gómez-Santillán, C., Fraire, H., Rangel-Valdez, N., Dorronsoro, B., Quiroz-Castellanos, M. (2023).** Comparative study of heuristics for the one-dimensional bin packing problem. Studies in Computational Intelligence, pp. 293–305. DOI: 10.1007/978-3-031-28999-6_19.
30. **Hussain, A., Aleem, M. (2018).** GoCJ: google cloud jobs dataset for distributed and cloud computing infrastructures. Data, Vol. 3, No. 4, pp. 38. DOI: 10.3390/data3040038.
31. **Feitelson, D. G., Nitzberg, B. (1995).** Job characteristics of a production parallel scientific workload on the NASA Ames iPSC/860. Lecture Notes in Computer Science, pp. 337–360. DOI: 10.1007/3-540-60153-8_38.
32. **HPC2N. (n.d.).** High performance computing center north (HPC2N) is a national center for scientific and parallel computing. https://www.hpc2n.umu.se/.

33. **Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y. P., Auger, A., Tiwari, S. (2005).** Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL report, pp. 1–50.
34. **Calheiros, R. N., Ranjan, R., De-Rose, C. A., Buyya, R. (2009)**. CloudSim: A novel framework for modeling and simulation of cloud computing infrastructures and services. ArXiv. DOI: 10.48550/ARXIV.0903.2525.
35. **Wang, S., Wang, L., Liu, M., Xu, Y. (2013).** An enhanced estimation of distribution algorithm for solving hybrid flow-shop scheduling problem with identical parallel machines. The International Journal of Advanced Manufacturing Technology, Vol. 68, No. 9-12, pp. 2043–2056. DOI: 10.1007/s00170-013- 4819-y.
36. **Liu, F., Zhang, X., Zou, F., Zeng, L. (2009).** Immune clonal selection algorithm for hybrid flow-shop scheduling problem. Chinese Control and Decision Conference. DOI: 10.1109/ccdc.2009.5194868.
37. **Xu, Y., Wang, L., Zhou, G., Wang, S. (2011).** An effective shuffled frog leaping algorithm for solving hybrid flow-shop scheduling problem. Lecture Notes in Computer Science, pp. 560–567. DOI: 10.1007/978-3-642-24728-6_76.
38. **Zhang, L., Li, K., Li, C., Li, K. (2017).** Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems. Information Sciences, Vol. 379, pp. 241–256. DOI: 10.1016/j.ins. 2016.08.003.
39. **Abdullahi, M., Ngadi, M. A., Abdulhamid, S. M. (2016).** Symbiotic organism search optimization based task scheduling in cloud computing environment. Future Generation Computer Systems, Vol. 56, pp. 640–650. DOI: 10.1016/j.future.2015.08.006.
40. **Elaziz, M. A., Xiong, S., Jayasena, K., Li, L. (2019).** Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution. Knowledge-Based Systems, Vol. 169, pp. 39–52. DOI: 10.1016/j. knosys.2019.01.023.
41. **Zuo, X., Zhang, G., Tan, W. (2014).** Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud. IEEE Transactions on Automation Science and Engineering, Vol. 11, No. 2, pp. 564–573. DOI: 10.1109/tase.2013.2272758.
42. **Mirjalili, S., Lewis, A. (2016).** The whale optimization algorithm. Advances in Engineering Software, Vol. 95, pp. 51–67. DOI: 10.1016/j.advengsoft.2016.01.008.
43. **Wang, G. (2016).** Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. Memetic Computing, Vol. 10, No. 2, pp. 151–164. DOI: 10.1007/ s12293-016-0212-3.
44. **Mirjalili, S. (2015).** The ant lion optimizer. Advances in Engineering Software, Vol. 83, pp. 80–98. DOI:10.1016/j.advengsoft.2015. 01.010.

Article received on 31/01/2024; accepted on 24/04/2024.
**Corresponding author is Jessica González-San-Martín.*

ISSN 2007-9737

Design of Routes for Collaborative Robots in the Automobile Painting Process through a Comparison of Perturbative Heuristics for Iterated Local Search

Lucero Ortiz-Aguilar[1,*], Luis Angel Xoca-Orozco[2], Juan de-Anda-Suárez[3]

[1] Tecnológico Nacional de México,
Instituto Tecnológico Superior de Purísima del Rincón,
División de Ingeniería en Sistemas Automotrices,
Mexico

[2] Tecnológico Nacional de México,
Instituto Tecnológico Superior de Purísima del Rincón,
División de Ingeniería Bioquímica,
Mexico

[3] Tecnológico Nacional de México,
Instituto Tecnológico Superior de Purísima del Rincón,
División de Ingeniería Electromecánica,
Mexico

{luis.xo, juan.ds}@purisima.tecnm.mx, ldm_oa@hotmail.com

Abstract. The automotive manufacturing industry faces diverse challenges, including designing new parts, optimizing schedule times, developing aerodynamic car designs, refining painting processes, and advancing autonomous driving. The car painting process is a new optimization area in the computational context due to the complexity and damage caused by various factors. Our research focuses on designing a robotic arm with five degrees of freedom that operates in a two-dimensional plane and is integrated with metaheuristics for path optimization. Our methodology consists of defining and limiting the problem, analyzing requirements, designing the robotic arm, implementing routes, and conducting tests. For the design of data instances, the Methodology proposed by [42] was used in this work. Subsequently, a pool of perturbation heuristics and an iterated local search algorithm are used, which allowed us to design the best combination of heuristics that can provide a competitive solution to the problem of route design for the robotic arm in the automobile painting process. This study includes a comprehensive review of related work, theoretical concepts, and the application of metaheuristics. The results highlight the effectiveness of the proposed heuristics, with the K-OPT heuristic demonstrating superior performance. Statistical tests confirm the significance of the differences among the heuristics. This paper concludes with insights into future research directions, emphasizing the importance of safety practices and Industry 4.0 technology in mitigating health risks associated with the automotive painting process.

Keywords. Iterated local search, routing design, heuristics.

1 Introduction

In the automotive manufacturing industry, there are different problems to solve, such as the design of new parts, optimization schedule times, new aerodynamic car designs, painting processes, and autonomous driving, among others.

Algorithm 1 Algorithm construction of robotic arm TSP solution

Require: $n \geq 0$, $FF \leftarrow$ Fitness Function
1: $S \leftarrow$ empty solution
2: route $\leftarrow$ empty list of size n
3: **for** $i \leftarrow 0$ **to** $n-1$ **do**
4: route$[i] \leftarrow$ random value between 0 and $n-1$
5: **end for**
6: $S \leftarrow \mathrm{set}(\text{route})$
7: $S \leftarrow \mathrm{evaluateSolution}(S, FF)$
8: **return** S

Algorithm 2 Two swap heuristic (h_1)

Require: $n \geq 1$, $SI \leftarrow$ Initial Solution, $FF \leftarrow$ Fitness Function
1: solutionB $\leftarrow \mathrm{copy}(SI)$
2: index $\leftarrow \mathrm{random}(0, n-1)$
3: index2 $\leftarrow \mathrm{random}(0, n-1)$
4: value $\leftarrow$ solutionB.[index2]
5: solutionB.[index2] $\leftarrow$ solutionB.[index]
6: solutionB.[index] $\leftarrow$ value
7: solutionB $\leftarrow \mathrm{evaluateSolution}(\text{solutionB}, FF)$
8: **return** $\mathrm{criteria\,.\,toReplace}(SI, \text{solutionB})$

The initial heuristic involves a double swap mechanism, requiring selecting two cities or nodes. These chosen cities exchange positions, creating a novel solution [46]. The algorithm is shown in 2.

Specifically, the painting process is one of the most common occurrences during car manufacturing and its use, as it becomes worn or damaged due to various factors. In the first case, when it comes to painting a car from the beginning, it is done with collaborative robots that work on specific work areas in most manufacturing companies.

The second case involves handmade workshop procedures with tools such as a compressor, airbrush, etc. For task assignment problems in collaborative robots, different algorithms help optimize the route and avoid possible collisions [67]. However, in most workshops, the car painters do not have performance methodologies that improve their work.

State of the art in metaheuristics, according to [23], has been applied to different problems such as scheduling [11], manufacturing and production, water management, oil and energy, traffic control, and among others.

Nevertheless, the actual problems in the industry are generally oriented to health, water, housing, education, and electromobility, and different artificial intelligence techniques can solve these problems. The most popular methods in recent research are neural networks [70], deep Learning [62], meta-learning [22], and others.

Therefore, there is a lot of research in the state of the art about the comparison of solving benchmark problems, artificial instances, or any other kind of problem [12, 27].

With these two contexts, metaheuristics can help solve industry problems, such as designing a route for collaborative robots or robots that work individually [28].

This work proposes a solution applicable to traditional painting workshops by designing a robotic arm with five degrees of freedom that can work in an x and y plane.

Our project aims to help the car painting process, where prolonged exposure to chemical paints causes serious human health problems (see section 3). Our proposal pays in the following points:

– Design of a robotic arm with five degrees of freedom adopted in a two-dimensional plane. The above is because, in the real application, it is easier and cheaper to implement.
– Integration of metaheuristics to design routes in the robot and parts.
– Methodology for recognizing parts and generating an optimized trajectory for your painting process.

The cost-benefit of the implementation of a Robotic Arm design and software for automotive painting processes are important since it covers the following aspects:

Algorithm 3 Crossover heuristic (h_2)

Require: $n \geq 1$, solutionA $\leftarrow$ Initial Solution, $FF \leftarrow$ Fitness Function

solutionB $\leftarrow \text{copy}(\text{solutionA})$
parent1 $\leftarrow$ solutionA.body
$n \leftarrow \text{len}(\text{parent1})$
parent2 $\leftarrow \text{sample}(\text{range}(0, n), n)$
$k \leftarrow \text{random}(0, n-1)$
Index $\leftarrow$ []
for i in $\text{range}(n)$ **do**
    if i != k **then**
        $\text{Index} \,.\, \text{append}(i)$
    end if
end for
$son \leftarrow$ []
for i in range(n) **do**
    if i ¿= n - len(Index) **then**
        $\text{son} \,.\, \text{append}(\text{parent1}[\text{Index}[i]])$
    else
        $\text{son} \,.\, \text{append}(-1)$
    end if
end for
$f \leftarrow$ []
for elem in parent2 **do**
    if elem not in son **then**
        $f \,.\, \text{append}(\text{elem})$
    end if
end for
for i in range(n) **do**
    if son[i] == -1 **then**
        $\text{son}[i] \leftarrow f \,.\, \text{pop}(0)$
    end if
end for
solutionB $\leftarrow \text{set}(\text{son})$
solutionB $\leftarrow \text{evaluateSolution}(\text{solutionB}, FF)$
return $\text{criteria} \,.\, \text{toReplace}(\text{SI}, \text{solutionB})$

– **Hardware and Software Acquisition Costs:** The physical implementation of the prototype is much less expensive than that of a commercial robotic arm. Likewise, it contains the sensors, controllers, and materials necessary for its correct operation.
– **Installation and Start-up Costs:** The maximum costs for installation and testing range from $100 to $150 dollars, which makes it a viable option that is quite competitive with others that cost around $5,000.
– **Use of Optimization Techniques**. Optimization techniques can be used within the complete painting process to design the routes the robot must follow, decreasing the cost and time of this task.

The expected benefits of implementing the system include labor savings, increased productivity, reduced waste, improved quality, and occupational safety. The rest of this article is structured in 6 sections described below.

An extensive review of relevant literature was reported in the related works section 2. The theory section 3 presented main concepts and theoretical definitions, while the methodology section 5 detailed the procedures used in the research.

The results section 6 showed the findings obtained from the implemented methodology. Finally, in the conclusions and future work section 7, an analysis was carried out based on the results collected throughout the experimentation with metaheuristics.

2 Related Work

2.1 Car Painting Problem

An important step in the automotive production system is the aesthetic finish of the vehicle paint, which changes for the different models and colors of production and the availability and quantity of paint ovens. Consequently, the problem of scheduling and sequencing vehicle models that minimize the waiting problem in the painting system arises.

State of the art has addressed different lines of research for the solution of the vehicle painting problem: The first focuses on the general aspects of the problem in the context of painting times and model sequences and poses the problem as an optimization system, where Metaheuristic algorithms solve the scheduling [15]; Secondly, the research focuses on minimizing the error corrections in the painting process by studying the nozzle and stroke size [49]; thirdly, the literature proposes that the solution lies in the

Algorithm 4 Nearest neighbor heuristic (h_3)

Require: $n \geq 1$, **solutionA** $\leftarrow$ Initial Solution, $FF \leftarrow$ Fitness Function, matrix $\leftarrow$ cost Matrix

```
solutionB ← copy . deepcopy(solutionA)
parent1 ← solutionA.body
c ← random. randint(0, len(parent1) − 1)
ic ← parent1. index(c)
rowt ← matrix[c]
rowOrderedWithIndex ← []
for i, value in enumerate(rowt) do
    rowOrderedWithIndex . append((i, value))
end for
rowOrderedWithIndex . sort(rowt)
orderedIndex ← []
for index, in rowOrderedWithIndex do
    orderedIndex . append(index)
end for
nc ← orderedIndex[1]
for i, in enumerate(parent1) do
    if parent1[i] == nc then
        inc ← i
        break
    end if
end for
c1 ← parent1[ic + 1]
parent1[ic + 1] ← nc
parent1[inc] ← c1
solutionB ← set(son)
solutionB ← evaluateSolution(solutionB, FF)
return criteria . toReplace(SI, solutionB)
```

robot trajectories and the painting sequence, and generates a new optimization algorithm by resampling and speed calibration [2, 8].

In another perspective, the painting problem is manipulated by a set of nodes in the finishing and painting process network. In the network, each process configures a certain task; however, from the point of view of process planning and optimization, it needs to start from an initial point [32, 24].

To solve the optimal scheduling problem, research has adapted the concept of the Voronoi diagram, which consists of separating the space around the key points of the painting process. Consequently, Voronoi diagrams allow the creation of an environment of efficiency and optimization that avoids wasting time and speeds up the coverage of the painting process [55].

2.2 Collaborative Robotic Problem

In an automotive industrial system, one of the fundamental paradigms is Industry 4.0, where we highlight the concept of collaborative robotics. Using graph theory, research has addressed the issues arising from robot collaboration in a paint-finishing environment [58].

Accordingly, the robots in the collaborative network are represented as nodes, whereas the edges are the process connections between the robots. Tackling the problems derived from collaborative robotics through graphs obtains different benefits: it allows studying the dynamics of the collaborative network, the distribution of the nodes and their edge connection, process synchronization, and coordination of the primary tasks [59, 57, 66].

To study the dynamics of the collaborative robot ensemble, the Euler-Lagrange formulation allows the mathematical modeling of the movements in the robotic arms. In a general context, Euler-Lagrange considers the principle of minimum energy of the event; this approach leads directly to an optimization in the movements of the painting and finishing processes of the vehicles [71, 13].

An aspect to highlight of Euler-Lagrange is the pressure of the dynamic models since it is based on the minimum action and allows the deterministic design of electronic controllers that make the automotive processes more efficient [60, 31].

In comparison, different techniques have addressed the problem of coordination and optimization of collaborative robotic networks: research in the design of controllers that make the processes in the network more efficient, adaptation of the small gain theorem, which analyzes the steady states in the control system under a linear dynamics approach; application of shrinkage analysis dedicated to the treatment of nonlinear control problems [68, 69, 33].

In summary, research in a collaborative robotics environment considers three elements of its behavior: simple control, linear control, and nonlinear control.

Algorithm 5 K-OPT (h_4 and h_5)

Require: $n \geq 2$, solutionA $\leftarrow$ Initial Solution, $FF \leftarrow FitnessFunction$

solutionB $\leftarrow \mathrm{copy}$(solutionA)
son $\leftarrow$ solutionB.body
ind $\leftarrow \mathrm{random.sample}(\mathrm{len}(\text{son}), n)$
$\mathrm{random.shuffle}$(ind)
sonCopy $\leftarrow$ son[:]
for i **in** range(self.n) **do**
    son[ind[i]] $\leftarrow$ sonCopy[ind[$(i+1)\%n$]]
end for
solutionB $\leftarrow \mathrm{set}$(son)
solutionB $\leftarrow \mathrm{evaluateSolution}$(solutionB, FF)
return $criteria.toReplace(SI, solutionB)$

Algorithm 6 Iterated Local Search (ILS)

Require: inSol $\leftarrow$ Initial Solution , maxI $\leftarrow$ MaxIterations

bestSol $\leftarrow$ inSol
currentSol $\leftarrow$ inSol
iteration $\leftarrow 0$
while iteration $<$ maxI **do**
    perturbedS $\leftarrow \mathrm{perturb}$(currentSol)
    localOptima $\leftarrow \mathrm{localSearch}$(perturbedSol)
    if $\mathrm{isImprovement}$(localOptima, bestSol) **then**
        bestSol $\leftarrow$ localOptima
    end if
    currentSol $\leftarrow \mathrm{diversify}$(localOptima)
    iteration $\leftarrow$ iteration $+ 1$
end while
return bestSol

The points mentioned above point to the complexity and diversity of problems and stochastic systems that collaborative robotics represents.

2.3 Health Risks During the Car Painting Process

The industry dedicated to vehicle painting plays a crucial role in enhancing the aesthetic appeal and providing protection to automobiles; however, this process entails significant health risks for workers exposed to the chemicals involved.

According to the World Health Organization (WHO), occupational exposure to chemicals found in paints and solvents affects approximately 11% of workers globally, including those exposed to organic solvents present in paints and varnishes [65].

These products often contain volatile organic compounds (VOCs) such as benzene, toluene, and xylene, which have been associated with mutagenic, carcinogenic, and teratogenic effects [43, 44, 48], as well as neurological disorders [61, 3, 36]. The International Agency for Research on Cancer (IARC) has classified some of the chemical components present in automobile paints as possible carcinogens [63], with benzene being linked to the development of leukemia.

Several reports have documented health issues experienced by automotive industry workers concerning their exposure to chemicals present in paints. For instance, Hammond et al. (2005) [20] report an increase in the incidence of asthma and chronic obstructive pulmonary disease, as well as an elevation in allergy-related symptoms such as eye and nose irritation, sinusitis, cough, and even heartburn.

Painters using aerosols have been observed to experience symptoms including excessive tearing, persistent cough, and short-term memory loss, along with a higher prevalence of respiratory symptoms, corneal opacity, and dry skin compared to the control group [40].

Furthermore, the risk is not confined solely to exposure to volatile organic compounds (VOCs), as it can also arise from contact or inhalation of metals such as cadmium, chromium, and nickel present in some paints. These metals can cause acute and chronic poisoning, as well as an increased risk of cancer [30, 39]. The primary Chemical Components Found in Paints Intended for the Automotive Industry and Their Health Effects are:

1. **Cadmium.** Acute and chronic poisonings, accumulation in kidneys, chronic decrement in renal, pulmonary, and hepatic function, causing ulcerations and perforations of the nasal septum, chronic bronchitis, decreased lung function, pneumonia, and other respiratory

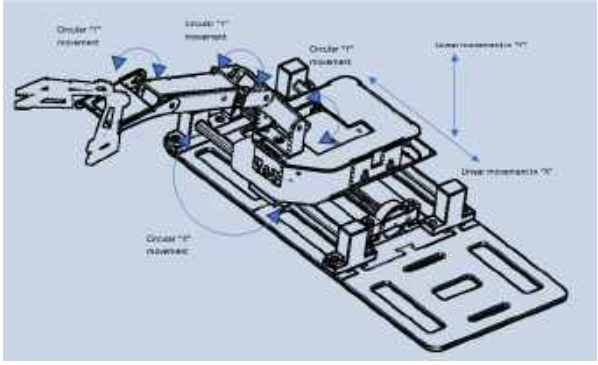


Fig. 1. Robotic arm design for the prototype. Source: own design

effects [6]. It exhibits impacts on bones, the respiratory system, the endocrine system, and the reproductive system [5]. Classified as carcinogenic to humans [53, 14].

2. **Chromium**. Allergic reactions and skin rashes, in addition to nasal irritation and bleeding [14]. Carcinogenic properties, hemolysis, and organ failure [38]. Genotoxic, toxic to reproduction and development [17].

   Weakening of the immune system, damage to the kidneys and liver, respiratory problems, disruption of genetic material, stomach discomfort and ulcers, and lung cancer [47, 56, 4].

3. **Nickel**. Systemic, immunological, neurological, reproductive, developmental distortion, and carcinogenic negative effects [9]. Accumulation in the kidney, inflammation of bronchioles, alveolar congestion, hyperplasia of alveolar cells [19].

   It may cause dizziness, pulmonary embolism, and respiratory failure [47]. It also leads to congenital disabilities, asthma, chronic bronchitis, heart disorders, and allergic reactions such as skin rashes.

   Associated with various types of lung, nasal, laryngeal, and prostate cancers. Induce various side effects, including allergies, cardiovascular and renal diseases, pulmonary fibrosis, and lung and nasal cancer. Epigenetic alterations are observed affecting genome level [16].

4. **Benceno**. Carcinogenic, causing cancer of the esophagus, ovaries, testicles, colon, and kidneys, as well as acute myeloid leukemia [52]. Induces dysbiosis of intestinal microbiota and metabolic disorders in mice [45, 18].

   Additionally, it is associated with atopic dermatitis, irritant contact dermatitis, allergic dermatitis, neoplasms, infections, skin irritation, hypersensitivity, mucosal irritation, rash, redness, skin swelling, allergic hypersensitivity reactions, cutaneous melanoma, non-melanoma skin cancer, dry skin, itching, rash, red itchy blisters, and burns [72, 64, 37].

5. **Tolueno**. Neuronal inhibitor associated with progressive damage to the central and peripheral nervous system and memory loss. It can lead to neurological dysfunction and hematological damage. It directly affects the central nervous system, causing euphoria, confusion, depression, headache, dizziness, hallucinations, seizures, ataxia, stupor, and coma [35].

   It may cause leukoencephalopathy and psychosis, as well as behavioral and functional abnormalities such as decreased memory capacity, cognitive impairment, and symptoms resembling depression [50].

6. **Xileno**. Neurotoxic effects, hepatotoxicity, and nephrotoxicity, leading to the formation of toxic intermediate and end products akin to those of benzene [21].

   It may cause nausea, headache, a sense of 'euphoria,' dizziness, weakness, irritability, vomiting, slow reaction time, confusion, clumsiness, difficulty speaking, loss of balance, ringing in the ears, drowsiness, loss of consciousness, and anemia. Results in fetotoxic effects such as delayed ossification [7, 26].

The reports above provide a clear and concerning perspective on the health hazards workers face exposed to chemicals in paints used in the automotive industry. These risks range from acute effects such as skin and respiratory tract irritation to more severe consequences, including

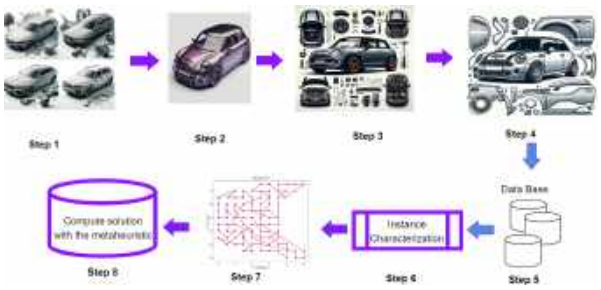


Fig. 2. Route design methodology for collaborative robots in automobile painting (source: own creation)

the development of chronic respiratory diseases and an increased risk of cancer. Adopting appropriate safety practices, adequate training, and using protective equipment are crucial to minimize these risks and preserve the health of workers in this industry.

Furthermore, it is relevant to develop strategies to reduce human contact, and one of these strategies is introducing industry 4.0 technology, which enables remote execution of activities.

3 Theory

3.1 Heuristics and Metaheuristics

In our study, we employed the approach outlined by Ortiz et al. [41], which employs various frameworks for managing Constraint Satisfaction Problems (CSP). An initial solution is generated, refined, and applied perturbative heuristics.

A simple definition of CSP can be found in [42, 41], defined with $V = v_1, v_2, \ldots, v_n$ a set of variables, $W = w_1, w_2, \ldots, w_m$ a set of values of each variable and $R = r_v1, r_v2, \ldots, r_{vn}$ a set of restrictions for each variable given.

3.1.1 Heuristics

A simple definition given by [51] is "*Heuristic approach is based on trial and error to find or discover solutions to the problems*". Two types of heuristics are most commonly used in metaheuristics: construction and perturbation. According to those mentioned earlier, we define construction and perturbative heuristics used in this work. According to [29], two main techniques for building Traveling Salesman Problem (TSP) solutions are greedy heuristic or aleatory and the formal definition of TSP can be found in [25]. In this investigation, the aleatory heuristic was chosen to make the initial solutions (see Algorithm 1) and five perturbative heuristics described in algorithms 2, 3.

The second heuristic is the **Crossover heuristic (h_2)**. It necessitates the use of two solutions. A starting solution (Solution A) is provided in the initial scenario, and a second solution (Solution B) is created using a different heuristic.

An arbitrary cutting point is chosen, where the first portion from Solution A forms the basis of the first offspring solution (SS1), complemented by the second segment from Solution B. The second offspring solution (SS2) is created by the initial part of Solution B combined with the latter segment of Solution A (see Algorithm 3).

The third heuristic, the **Nearest Neighbor (h_3)**, involves selecting a random point from an initial solution. Subsequently, the heuristic identifies the city closest to this chosen point (see Algoritmn 4).

The fourth heuristic, known as **K-OPT (h_4)**, operates by choosing three variables and exchanging their respective values among themselves. If the exchange is not feasible, the initial solution is returned. In the TSP, this method is called K-interchange. Finally, the last heuristic is **4-OPT (h_5)**: It has the same functionality as K-swap with k=4.

3.1.2 Metaheuristics

The metaheuristics are used to solve problems intelligently by choosing the best from a larger number of available solutions [23]. A metaheuristic algorithm lets one search for optimal solutions to a particular problem with some restrictions.

This searching process can involve some operators such as selection, mutation, improvement, crossover, and, in specific cases, a set of rules or mate mathematical equations that provide a guide during multiple iterations. These iterations are carried out until the solution found meets some criterion [23].

Table 1. Description of cars used for 3D modeling

No.	Name	Points	Length Path	No.	Name	Points	Length Path
1	AUDI-R8-CAPO-2024	90	93.0	30	dodge-charger-1969-CARROCERIA	165	192.0
2	AUDI-R8-CARROCERIA-2024	174	175.4	31	dodge-charger-1969-PUERTAS	53	54.0
3	AUDI-R8-PARTE TRASERA-2024	78	81.0	32	FERRARI-CAPO	84	83.0
4	AUDI-R8-PUERTAS-2024	42	41.0	33	FERRARI-CARROCERIA	250	309.0
5	AVEO-SEDAN-CAPO	53	52.0	34	FERRARI-PUERTAS	38	37.0
6	AVEO-SEDAN-CARROCERIA	136	176.0	35	FORD F150 RAPTOR-CAPO	107	107.0
7	AVEO-SEDAN-PARTE TRASERA	58	57.0	36	FORD F150 RAPTOR-PUERTA TRASERA	55	54.0
8	AVEO-SEDAN-PUERTAS	28	27.0	37	FORD F150 RAPTOR-PUERTAS	71	70.0
9	BUGATTI-CARROCERIA	41	44.0	38	FORD FIESTA MK3-4-CAPO	54	53.0
10	BUGGATI CHIRON-CAPO	35	34.0	39	FORD FIESTA MK3-4-CARROCERIA	120	134.0
11	BUGGATI CHIRON-CARROCERIA	189	215.8	40	FORD FIESTA MK3-4-PUERTA TRASERA	86	85.4
12	BUGGATI CHIRON-PUERTA TRASERA	45	48.0	41	FORD FIESTA MK3-4-PUERTAS	67	66.0
13	BUGGATI CHIRON-PUERTA	62	61.0	42	FORD GT40 -3-CAPO	53	53.0
14	CADILLAC ELR-CAPO	48	47.0	43	FORD GT40 -3-CARROCERIA	82	83.6
15	CADILLAC ELR-CARROCERIA	52	54.0	44	FORD GT40 -3-PUERTAS	103	103.0
16	CADILLAC ELR-PUERTA TRASERA	28	27.0	45	FORD MUSTANG -3-CAPO	83	82.0
17	CADILLAC ELR-PUERTA	66	68.6	46	FORD MUSTANG -3-CARROCERIA	82	83.0
18	camaro-z-28-1969-CAPO	132	131.0	47	FORD MUSTANG -3-PUERTAS	107	120.0
19	camaro-z-28-1969-CARROCERIA	101	104.0	48	FORD MUSTANG 1965 -4-CAJUELA	36	35.0
20	camaro-z-28-1969-PUERTA	64	65.0	49	FORD MUSTANG 1965 -4-CAPO	76	75.0
21	Chevrolet-Camaro-model-CAPO	56	55.0	50	FORD MUSTANG 1965 -4-CARROCERIA	78	77.0
22	Chevrolet-Camaro-model-CARROCERIA	44	49.0	51	FORD MUSTANG 1965 -4-PUERTAS	36	35.0
23	chevrolet-camaro-model-PARTE TRASERA	24	23.0	52	FORD MUSTANG 1967 -4-CAPO	30	29.0
24	Chevrolet-Camaro-model-PUERTAS	79	81.5	53	FORD MUSTANG 1967 -4-CARROCERIA	64	63.0
25	DATSUN 1500 PICK UP-CAPO	48	47.0	54	FORD MUSTANG 1967 -4-PUERTA TRASERA	18	17.0
26	DATSUN 1500 PICK UP-CARROCERIA	66	68.0	55	FORD MUSTANG 1967 -4-PUERTAS	34	33.0
27	DATSUN 1500 PICK UP-PUERTA TRASERA	40	39.0	56	FORD RANGER -3-CAJUELA	18	17.0
28	DATSUN 1500 PICK UP-PUERTA	41	40.0	57	FORD RANGER -3-CARROCERIA	84	83.0
29	dodge-charger-1969-CAPO	47	46.0	58	FORD RANGER -3-PUERTAS	66	69.0

The Iterated Local Search algorithm (ILS) was used in the experimentation. This metaheuristic was proposed by Lourenço et al. [34], which constructs or modifies a solution through an embedded heuristic. The generated solutions are better than randomly generated or altered ones. This algorithm aims to intensify the initial solution by exploring its neighbors. The ILS algorithm is depicted in 6, taken from Talbi El-Ghazali [54].

3.2 Car Painting Problem

The car painting problem has had different variations, and according to [10], the problem above derives from the Paint Shop Problem for Words. In a paint shop, cars go through two painting steps. The first paint step applies a coat of filler paint, and the second one applies a coat of base paint depending on the final color of the car.

According to the type of filling, they are divided into white and black for each car model and each base color. Each of the two painting steps is done using separate spray nozzles, which must be cleaned before a new color can be applied, which can be expensive.

The above results in the definition of the Multi-Car Multi-Color Paint Shop Problem, whose objective is to find a color assignment in the sequence that minimizes the number of color changes for the fill paint nozzle and the paint nozzle base and that also satisfies the color demand required for the different models [10].

3.3 Trajectory Design in Car Painting Problem

With the previous context, this work proposes the design of routes for a robotic arm that can execute within a controlled environment. For our problem, we consider that a vehicle can be cleared into

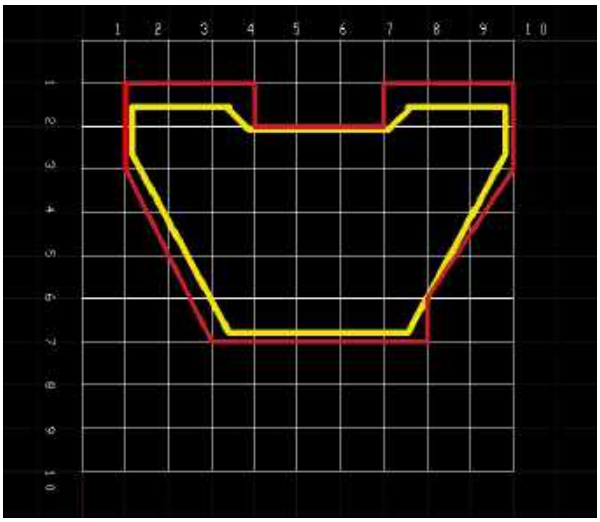


Fig. 3. 3D vehicle hood with mesh (own source)

a w number of pieces. So, each vehicle V has several parts, each denoted as $v_0 \dots v_w$. Each piece v has a set of points P denoted as $p_1 \dots p_n$, which allows generating a matrix of costs M_C and adjacencies M_A.

With the above, we seek to create a permutation of the P points for each piece v so that it does not go through both places simultaneously. The mathematical expression corresponding to the calculated distance of the points P of each of the pieces is defined as:

$$\text{distance} = \sum_{i=0}^{n-1} \text{dDistancesC}[f_i][f_{i+1}], \qquad (1)$$

where:

- distance represents the calculated distance.
- $\text{dDistancesC}[f_i][f_{i+1}]$ represents the distance between nodes f_i and f_{i+1}.
- n is the length of the list f.

3.4 Robotic Arm Design

The arm design has two types of movements: circular and linear. Each modeled part was assembled using the Autodesk Inventor Professional 2023 software.

Using the restricted tools, their function coincides with the central axes of the cylindrical figures with which their same axes were joined.

Afterward, a matching function placed each piece glued to another. Once these assemblies were in place, the leveling function was used to overlap the bases of the robotic arm to match the upper parts at the same point as the fixation tool to prevent the piece from moving with the other assemblies. The final prototype can be seen in Figure 1. This work focuses exclusively on the development of metaheuristics for designing routes for the robotic arm; therefore, subsequent research will explain its design.

4 Methodology

The methodology developed in this work is described in the next sections.

- **Definition and Limitation of the Problem.** The different cars will be chosen and designed in a CAD sketch in this stage. The type of car and the number of parts each drawing will include will be defined. We define the principal parts such as the roof, doors, bonnet, and bumper.

  Subsequently, all completed designs will be collated to form a database of CAD models. This would represent steps 1 and 2 shown in figure 2. The cars considered for the database were selected among the most common models on the global market, such as Mazda, Ford, etc.
- **Requirements Analysis.** In Figure 2, steps 3 and 4 refer to the breakdown of the cars in their different views. Once each car instance has been obtained, the points of interest will be assigned to those through which the robotic arms must pass to paint each piece. With this information, a database will be made (step 5), which will subsequently be characterized by instances (step 6).
- **Design.** In the design phase, the robotic arms' routes will be designed to respect all the restrictions posed in the problem limitation phase using heuristic construction techniques.

Table 2. Shapiro Wilks statistics for each heuristic

Heuristic	Statistic	P-Value
h_1	0.721222043	3.54E-09
h_2	0.645808816	1.45E-10
h_3	0.652767837	1.91E-10
h_4	0.660536528	2.60E-10
h_5	0.665777683	3.22E-10

– **Implementation.** Once the routes have been designed, it will be validated which of them are feasible to be applied in a simulation (see Figure 2 in step 7 and step 8). A statistical comparison of the routes built will be made.

– **Tests.** In this phase, we move on to applying them either in a simulation or in a physical prototype. In addition, all tests and corrections will be made regarding the path of the robotic arms, either physically or through simulation.

5 Methodology

The methodology developed in this work is described in the next sections.

– **Definition and Limitation of the Problem.** The different cars will be chosen and designed in a CAD sketch in this stage. The type of car and the number of parts each drawing will include will be defined. We define the principal parts such as the roof, doors, bonnet, and bumper.

Subsequently, all completed designs will be collated to form a database of CAD models. This would represent steps 1 and 2 shown in figure 2. The cars considered for the database were selected among the most common models on the global market, such as Mazda, Ford, etc.

– **Requirements Analysis.** In Figure 2, steps 3 and 4 refer to the breakdown of the cars in their different views. Once each car instance has been obtained, the points of interest will be assigned to those through which the robotic arms must pass to paint each piece.

With this information, a database will be made (step 5), which will subsequently be characterized by instances (step 6).

– **Design.** In the design phase, the robotic arms' routes will be designed to respect all the restrictions posed in the problem limitation phase using heuristic construction techniques.

– **Implementation.** Once the routes have been designed, it will be validated which of them are feasible to be applied in a simulation (see Figure 2 in step 7 and step 8). A statistical comparison of the routes built will be made.

– **Tests.** In this phase, we move on to applying them either in a simulation or in a physical prototype. In addition, all tests and corrections will be made regarding the path of the robotic arms, either physically or through simulation.

6 Results

Based on the criteria outlined in sections 3 and 5, the car painting problem demands a dedicated data structure as presented in [41]. For this instance set, our approach leaned towards employing the MMA and LPH structures.

These were selected based on their relevance and applicability within the problem's framework, as detailed in the referenced work by Ortiz et al. (2023).

Different 3D models of vehicles were designed using the computer mechanical design software Inventor 3D. A set of 58 CAD designs was obtained according to various characteristics.

We grouped the cars according to the number of pieces counted, and the designs are described in table 1.

After selecting the vehicle 3D-CAD models, they were imported into Auto CAD to work in a 2D format. This is because if each piece has to be painted with a specific route, it needs to be in a two-dimensional plane to know the coordinate (x, y), where the robotic arm will be located.

Subsequently, a $20X20$ mesh was drawn in each piece, which let us draft our piece's contours to generate the trajectory coordinates. These coordinates help us determine which points are of

Table 3. Median for heuristics

	h_1	h_2	h_3	h_4	h_5		h_1	h_2	h_3	h_4	h_5
1	**131.6**	383.1	210.7	135.9	161.8	30	**308.7**	1283.9	710.0	383.7	462.7
2	**299.5**	1114.1	551.1	356.5	429.6	31	73.2	189.3	112.3	**68.4**	78.1
3	**112.8**	314.9	172.8	114.0	129.4	32	**117.8**	320.0	173.8	119.1	138.6
4	**50.9**	103.8	62.7	48.7	52.7	33	**570.0**	2363.6	1194.0	714.2	860.3
5	67.5	152.2	91.2	**64.6**	71.8	34	46.0	87.2	57.8	**43.5**	45.5
6	**249.0**	894.5	499.3	271.9	343.5	35	**158.2**	494.3	244.2	170.1	202.5
7	73.3	175.5	101.0	**72.0**	82.8	36	71.7	171.2	99.8	**67.3**	77.4
8	31.1	52.6	36.4	**30.4**	30.7	37	**97.6**	266.6	140.1	102.3	117.7
9	54.0	121.5	71.1	**51.9**	56.6	38	66.6	157.6	91.6	**65.3**	74.5
10	40.4	75.9	49.5	**40.3**	42.0	39	**194.6**	663.8	350.5	214.3	255.7
11	**346.2**	1379.5	745.9	438.4	529.5	40	**136.7**	413.4	220.9	140.7	165.4
12	**55.0**	126.4	76.9	**55.0**	59.8	41	96.2	251.9	144.7	**94.1**	112.0
13	79.8	199.4	114.7	**77.4**	92.2	42	64.9	154.2	89.8	**64.0**	72.9
14	58.2	127.6	74.9	**56.7**	62.4	43	125.2	374.9	215.3	**123.7**	145.0
15	**66.5**	161.4	93.4	67.3	73.3	44	**162.1**	500.1	284.7	166.7	200.7
16	**29.5**	52.7	35.8	30.9	30.8	45	116.3	315.2	168.4	**115.5**	138.5
17	92.2	238.3	126.8	**90.3**	102.3	46	**116.4**	342.1	185.5	118.8	139.2
18	**194.1**	661.2	328.1	228.4	273.4	47	**172.7**	582.3	330.4	185.8	229.7
19	**147.1**	477.0	256.7	162.0	191.0	48	41.7	83.5	54.1	**40.1**	42.0
20	97.6	271.7	143.8	**95.3**	108.3	49	102.6	270.0	145.7	**102.2**	120.8
21	71.4	164.8	94.0	**67.5**	77.4	50	**107.8**	302.1	166.6	109.5	125.1
22	56.1	125.8	77.9	**54.1**	58.7	51	42.0	82.8	54.3	**40.6**	41.8
23	**26.0**	42.4	31.0	26.2	25.7	52	**32.7**	58.0	39.1	**32.7**	33.8
24	**110.9**	328.2	180.4	112.0	134.7	53	83.5	214.6	125.6	**81.3**	96.4
25	57.7	127.4	76.2	**56.9**	62.7	54	18.4	24.8	19.2	18.7	**18.2**
26	94.4	248.8	140.3	**90.9**	103.5	55	40.7	74.2	48.1	38.9	**40.2**
27	48.7	100.8	63.7	**46.7**	50.6	56	18.4	24.4	18.6	18.7	**17.8**
28	50.4	105.1	59.6	**49.1**	52.6	57	**117.5**	347.5	183.8	123.4	142.8
29	59.3	123.9	76.5	**55.4**	60.7	58	**88.4**	221.0	123.4	88.5	101.3

interest and which must be left inside so that the piece is painted in its entirety by the robotic arm described in 3.4. An example of meshing is found in Figure 3. In Figure 3, the hood of our vehicle is shown in yellow, while red represents the contour that establishes the limits where our algorithm will trace the trajectories that will cover the hood.

With the help of our Cartesian coordinates application for Auto CAD loaded from the AP command, we will place the points that will be our coordinates in X and Y, which will be exported directly to an Excel file. This $.csv$ file was used to determine our routes and generate a matrix to program our algorithm.

Table 4. Heuristics ranks per test

	h_1	h_2	h_3	h_4	h_5
FT	1.7	5.0	4.0	**1.6**	2.8
AFT	1.7	5.0	4.0	**1.6**	2.8
QT	1.9	5.0	4.0	**1.5**	2.6

6.1 Heuristics Results

We will describe the parameter configuration for each construction heuristics and explain the statistical tests used to compare the experiments. The experimentation involved testing Instances on a laptop with the following specifications: an Intel® Core™ i3-1005G1 CPU clocked at 1.20 GHz, 8 GB of RAM, and a 64-bit Windows 11 operating system.

The methodology was implemented using Python, and the experiments were iterated 31 times. Each heuristic experiment had a stipulated limit of 100,000 function calls per instance.

For the statistical analysis, first, the Shapiro-Wilks statistical test was applied to check if the data had a normal distribution, and the results indicated that the data did not follow a normal distribution; therefore, the median was chosen as the statistical representative.

We use the benchmark proposed in previous sections with 58 instances tested with 31 runs. We show our results in Table 2. The Friedman (FT), Alienated Friedman (AFT), and Quade (QT) statistical tests were applied to distinguish the performance of the heuristic set.

We set $\alpha = 0.05$ and h_0 : there are no differences between the performance of the heuristics and h_a : there are differences between the performance of the heuristics. Table 4 shows the ranges obtained in the three statistical tests.

The results show that the best heuristic is h_4, based on the Friedman, Friedman Aligned, and Quade tests. It is important to remember that the no-free fight theorem mentions that each algorithm will perform differently according to the context.

Therefore, no algorithm is best for all types of problems [1]. The experimentation present in this work is based on the characteristics of the problem, and we report the results fairly (the same function calls); in this case, the best heuristic was h_4. It should be noted that the objective of this work is to use and present the effectiveness of the tools previously proposed by [41], where different problems of the Constraint Satisfaction Problem are addressed.

With the above, the generality of the design and solution methodology for various combinatorial issues continues to be demonstrated.

7 Conclusion and Future Work

In the present work, state-of-the-art research was reviewed, focusing specifically on the problem of programming and sequencing vehicle models to minimize times in the painting system.

In addition, the health risks faced by workers exposed to chemical agents present in automotive paints were researched.

After carrying out this state-of-the-art study, the problem is usually solved with different techniques in a very complex way and sometimes very expensive in terms of algorithms.

Therefore, this area of opportunity was identified, in addition to the future vision of continuing with the prototype and testing it with pieces that can be painted by it and scaled.

Methodologically, the current research reviews and exposes the application of the methodology proposed in [41] to optimize the scheduling of vehicle painting tasks, thus addressing existing CAD designs and achieving a route design for its future simulation.

Analyzing the results obtained from the experimentation provides information on the effectiveness of the heuristics used to address the automobile painting problem.

Utilizing a variety of statistical tests, among which are Shapiro-Wilks and the Friedman, Friedman Aligned, and Quade tests for general comparison, significant variations in the performance of the heuristics (h_1 to h_5) are observed.

Due to the data do not follow a normal distribution, the median was chosen as the representative measure for statistical analysis. In particular, the h_4 heuristic consistently

demonstrated superior performance in all statistical tests. Furthermore, this work reaffirms the applicability of the methodology proposed by Ortiz et al. (2023) by addressing different problems with combinatorial optimization constraints.

The findings contribute to a deeper understanding of the selection heuristics and the methodology's effectiveness in solving complex constraint satisfaction problems. In this case, the study problem was the design of routes for the painting process in automobiles.

Acknowledgments

We thank the Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCYT) and the Tecnológico Nacional de México/Instituto Tecnológico Superior de Purísima del Rincón for their time and support in the development of this research.

References

1. **Adam, S. P., Alexandropoulos, S. A. N., Pardalos, P. M., Vrahatis, M. N. (2019).** No free lunch theorem: A review. Approximation and optimization: Algorithms, complexity and applications, pp. 57–82. DOI: 10.1007/978-3-030-12767-1_5.
2. **Andulkar, M. V., Chiddarwar, S. S., Marathe, A. S. (2015).** Novel integrated offline trajectory generation approach for robot assisted spray painting operation. Journal of Manufacturing Systems, Vol. 37, pp. 201–216. DOI: 10.1016/j.jmsy.2015.03.006.
3. **Attarchi, M. S., Labbafinejad, Y., Mohammadi, S. (2010).** Occupational exposure to different levels of mixed organic solvents and colour vision impairment. Neurotoxicology and teratology, Vol. 32, No. 5, pp. 558–562. DOI: 10.1016/j.ntt.2010.05.003.
4. **Barceloux, D. G., Barceloux, D. (1999).** Chromium. Journal of Toxicology: Clinical Toxicology, Vol. 37, No. 2, pp. 173–194. DOI: 10.1081/CLT-100102418.
5. **Bernard, A. (2008).** Cadmium & its adverse effects on human health. Indian journal of medical research, Vol. 128, No. 4, pp. 557–564.
6. **Bradshaw, L. M., Fishwick, D., Slater, T., Pearce, N. (1998).** Chronic bronchitis, work related respiratory symptoms, and pulmonary function in welders in New Zealand. Occupational and Environmental Medicine, Vol. 55, No. 3, pp. 150. DOI: 10.1136/oem.55.3.150.
7. **Caro, J., Gallego, M., Montero, R. (2009).** Diferentes metodologías para la evaluación de riesgos originados por compuestos orgánicos volátiles en ambientes laborales. Seguridad y Medio Ambiente, Vol. 113, pp. 20–36.
8. **Chen, W., Zhao, D. (2013).** Path planning for spray painting robot of workpiece surfaces. Mathematical Problems in Engineering, Vol. 2013. DOI: 10.1155/2013/659457.
9. **Das, K. K., Das, S. N., Dhundasi, S. A. (2008).** Nickel, its adverse health effects & oxidative stress. Indian journal of medical research, Vol. 128, No. 4, pp. 412–425.
10. **Debevere, P., Sugimura, M., Parizy, M. (2023).** Quadratic unconstrained binary optimization for the automotive paint shop problem. IEEE Access. DOI: 10.1109/ACCESS.2023.3313102.
11. **Dokeroglu, T., Kucukyilmaz, T., Talbi, E. G. (2023).** Hyper-heuristics: A survey and taxonomy. Computers & Industrial Engineering, pp. 109815. DOI: 10.1016/j.cie.2023.109815.
12. **Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., Cosar, A. (2019).** A survey on new generation metaheuristic algorithms. Computers & Industrial Engineering, Vol. 137, pp. 106040. DOI: 10.1016/j.cie.2019.106040.
13. **Dong, F., Jiang, H., Qian, X., Yu, X. (2020).** Optimal motion planning for industrial robot based on the Euler-Lagrange equation. Journal of Intelligent & Robotic Systems, Vol. 98, No. 3, pp. 709–724. DOI: 10.1007/s10846-019-01050-0.

14. **Elonheimo, H. M., Mattila, T., Andersen, H. R., Bocca, B., Ruggieri, F., Haverinen, E., Tolonen, H. (2022).** Environmental substances associated with chronic obstructive pulmonary disease—A scoping review. International Journal of Environmental Research and Public Health, Vol. 19, No. 7, pp. 3945. DOI: 10.3390/ijerph19073945.

15. **Gasparetto, A., Vidoni, R., Pillan, D., Saccavini, E. (2012).** Automatic path and trajectory planning for robotic spray painting. ROBOTIK 2012, 7th German Conference on Robotics, pp. 1–6.

16. **Genchi, G., Carocci, A., Lauria, G., Sinicropi, M. S., Catalano, A. (2020).** Nickel: Human health and environmental toxicology. International journal of environmental research and public health, Vol. 17, No. 3, pp. 679. DOI: 10.3390/ijerph17030679.

17. **Gogoi, K., Manna, P., Dey, T., Kalita, J., Unni, B. G., Ozah, D., Baruah, P. K. (2019).** Circulatory heavy metals (cadmium, lead, mercury, and chromium) inversely correlate with plasma GST activity and GSH level in COPD patients and impair NOX4/Nrf2/GCLC/GST signaling pathway in cultured monocytes. Toxicology in Vitro, Vol. 54, pp. 269–279. DOI: 10.1016/j.tiv.2018.10.010.

18. **Guo, J., Kauppinen, T., Kyyrönen, P., Heikkilä, P., Lindbohm, M. L., Pukkala, E. (2004).** Risk of esophageal, ovarian, testicular, kidney and bladder cancers and leukemia among finnish workers exposed to diesel or gasoline engine exhaust. International Journal of Cancer, Vol. 111, No. 2, pp. 286–292. DOI: 10.1002/ijc.20263.

19. **Gupta, A. D., Patil, A. M., Ambekar, J. G., Das, S. N., Dhundasi, S. i. A., Das, K. K. (2006).** L-ascorbic acid protects the antioxidant defense system in nickel-exposed albino rat lung tissue. Journal of basic and clinical physiology and pharmacology, Vol. 17, No. 2, pp. 87–100.

20. **Hammond, S. K., Gold, E., Baker, R., Quinlan, P., Smith, W., Pandya, R., Balmes, J. (2005).** Respiratory health effects related to occupational spray painting and welding. Journal of occupational and environmental medicine, Vol. 47, No. 7, pp. 728–739. DOI: 10.1097/01.jom.0000165748.31326.e8.

21. **Haro-García, L., Vélez-Zamora, N., Aguilar-Madrid, G., Guerrero-Rivera, S., Sánchez-Escalante, V., Muñoz, S. R., Mezones-Holguín, E., Juárez-Pérez, C. (2012).** Alteraciones hematológicas en trabajadores expuestos ocupacionalmente a mezcla de benceno-tolueno-xileno (btx) en una fábrica de pinturas. Revista Peruana de Medicina Experimental y Salud Pública, Vol. 29, pp. 181–187.

22. **Hospedales, T., Antoniou, A., Micaelli, P., Storkey, A. (2021).** Meta-learning in neural networks: A survey. IEEE transactions on pattern analysis and machine intelligence, Vol. 44, No. 9, pp. 5149–5169. DOI: 10.1109/TPAMI.2021.3079209.

23. **Hussain, K., Mohd-Salleh, M. N., Cheng, S., Shi, Y. (2019).** Metaheuristic research: a comprehensive survey. Artificial intelligence review, Vol. 52, pp. 2191–2233. DOI: 10.1007/s10462-017-9605-z.

24. **Jiang, P., Yin, X. (2018).** Optimization of process planning in manufacturing networks based on genetic algorithm. Mathematical Problems in Engineering, Vol. 2018, pp. 1–12. DOI: 10.1155/2018/1571089.

25. **Jünger, M., Reinelt, G., Rinaldi, G. (1995).** The traveling salesman problem. Handbooks in operations research and management science, Vol. 7, pp. 225–330. DOI: 10.1016/S0927-0507(05)80121-5.

26. **Kandyala, R., Raghavendra, S. P. C., Rajasekharan, S. T. (2010).** Xylene: An overview of its health hazards and preventive measures. Journal of oral and maxillofacial pathology, Vol. 14, No. 1, pp. 1–5. DOI: 10.4103/0973-029X.64299.

27. **Khanduja, N., Bhushan, B. (2021).** Recent advances and application of metaheuristic algorithms: A survey (2014–2020).

Metaheuristic and evolutionary computation: algorithms and applications, pp. 207–228. DOI: 10.1007/978-981-15-7571-6_10.

28. **Kinast, A., Braune, R., Doerner, K. F., Rinderle-Ma, S., Weckenborg, C. (2022).** A hybrid metaheuristic solution approach for the cobot assignment and job shop scheduling problem. Journal of Industrial Information Integration, Vol. 28, pp. 100350. DOI: 10.1016/j.jii.2022.100350.
29. **Laporte, G., Semet, F. (2002).** Classical heuristics for the capacitated VRP. The vehicle routing problem, pp. 109–128. DOI: 10.1137/1.9780898718515.ch5.
30. **Lin, C. H., Lai, C. H., Peng, Y. P., Wu, P. C., Chuang, K. Y., Yen, T. Y., Xiang, Y. K. (2019).** Comparative health risk of inhaled exposure to organic solvents, toxic metals, and hexavalent chromium from the use of spray paints in Taiwan. Environmental Science and Pollution Research, Vol. 26, pp. 33906–33916. DOI: 10.1007/s11356-018-2669-8.
31. **Liu, Y., Chen, W., Huang, W. (2018).** Pressure-based optimal dynamic trajectory planning for robot manipulators via improved Euler-Lagrange equation. Journal of Intelligent & Robotic Systems, Vol. 89, No. 1, pp. 89–102. DOI: 10.1007/s10846-017-0711-1.
32. **Liu, Z. Y., Hu, H., Li, Z. Y., Ren, X. D. (2020).** An improved immune genetic algorithm for process planning optimization in manufacturing networks. Computational Intelligence and Neuroscience, Vol. 2020, pp. 1–13. DOI: 10.1155/2020/8883450.
33. **Liu, Z. Y., Hu, H., Li, Z. Y., Ren, X. D. (2020).** Shrinkage analysis for nonlinear networked control systems with an application to robotic manipulators. IEEE Transactions on Industrial Electronics, Vol. 67, No. 8, pp. 7153–7162. DOI: 10.1109/TIE.2019.2945518.
34. **Lourenço, H. R., Martin, O. C., Stützle, T. (2003).** Iterated local search. Handbook of Metaheuristics, Springer New York, Vol. 57, pp. 320–353. DOI: 10.1007/0-306-48056-5_11.
35. **Mahavar, S., Chaturvedi, A., Singh, A., Kumar, R., Dariya, S. S., Sharma, R. (2018).** Toluene poisoning presenting as bilateral basal ganglia haemorrhage. The Journal of the Association of Physicians of India, Vol. 66, No. 9, pp. 93–94.
36. **Mediouni, Z., Potherat, G., Barrere, X., Debure, A., Descatha, A. (2011).** Renal failure and occupational exposure to organic solvents: What work-up should be performed?. Archives of Environmental & Occupational Health, Vol. 66, No. 1, pp. 51–53. DOI: 10.1080/19338244.2010.506501.
37. **Márquez-Torres, A. M., Bojorque-Bojorque, L. M., Ortiz-Freire, G. E., Quito-Ochoa, P. Y. (2022).** Benceno y alteraciones crónicas dermatológicas en trabajadores de gasolineras. Salud, Ciencia y Tecnología, Vol. 2, No. S1, pp. 186. DOI: 10.56294/saludcyt2022186.
38. **Nasirzadeh, N., Mohammadian, Y., Dehgan, G. (2021).** Health risk assessment of occupational exposure to hexavalent chromium in iranian workplaces: A meta-analysis study. Biological Trace Element Research, Vol. 200, pp. 1–10. DOI: 10.1007/s12011-021-02789-w.
39. **Nduka, J. K., Kelle, H. I., Amuka, J. O. (2019).** Health risk assessment of cadmium, chromium and nickel from car paint dust from used automobiles at auto-panel workshops in Nigeria. Toxicology reports, Vol. 6, pp. 449–456. DOI: 10.1016/j.toxrep.2019.05.007.
40. **Ojo, T. O., Onayade, A. A., Afolabi, O. T., Ijadunola, M. Y., Esan, O. T., Akinyemi, P. A., Awe, O. O. (2020).** Work practices and health problems of spray painters exposed to organic solvents in Ile-Ife, Nigeria. Journal of Health Pollution, Vol. 10, No. 28, pp. 201208. DOI: 10.5696/2156-9614-10.28.201208.
41. **Ortiz-Aguilar, L., Yeovanna, H. A., Benitez, M., Rodriguez-Miranda, S., Mendoza-Vazquez, F. (2023).** A comparison between selection operators

heuristics of perturbation in CSP. Hybrid Intelligent Systems Based on Extensions of Fuzzy Logic, Neural Networks and Metaheuristics, pp. 365–377. DOI: 10.1007/978-3-031-28999-6_23.

42. **Ortiz-Aguilar, L., Calzada-Ledesma, V., de-Anda-Suárez, J., Bautista-Sánchez, R., Zapata-Gonzalez, N. (2022).** A comparison of replacement operators in heuristics for CSP problems. New Perspectives on Hybrid Intelligent System Design based on Fuzzy Logic, Neural Networks and Metaheuristics, pp. 335–353. DOI: 10.1007/978-3-031-08266-5_22.
43. **Palma, M., Briceño, L., Idrovo, Á. J., Varona, M. (2015).** Evaluación de la exposición a solventes orgánicos en pintores de carros de la ciudad de Bogotá. Biomédica, Vol. 35, No. SPE, pp. 66–76. DOI: 10.7705/biomedica.v35i0.2268.
44. **Park, J., Shin, K. S., Kim, Y. (2010).** Occupational reproductive function abnormalities and bladder cancer in Korea. Journal of Korean medical science, Vol. 25, No. Suppl, pp. S41–S45. DOI: 10.3346/jkms.2010.25.S.S41.
45. **Quintero, M. Z., Ruiz, A. M. V., Trujillo, I. C. O. (2009).** Efecto genotóxico y mutagénico de contaminantes atmosféricos. Medicina UPB, Vol. 28, No. 1, pp. 33–41.
46. **Rego, C., Gamboa, D., Glover, F., Osterman, C. (2011).** Traveling salesman problem heuristics: Leading methods, implementations and latest advances. European Journal of Operational Research, Vol. 211, No. 3, pp. 427–441. DOI: 10.1016/j.ejor.2010.09.010.
47. **Rodríguez-Heredia, D. (2017).** Intoxicación ocupacional por metales pesados. Medisan, Vol. 21, No. 12, pp. 3372–3385.
48. **Rubin, S. M., Clapp, R. (2012).** Patterns of mortality among wisconsin uniroyal tire manufacturing workers. NEW SOLUTIONS: A Journal of Environmental and Occupational Health Policy, Vol. 21, No. 4, pp. 603–620. DOI: 10.2190/NS.21.4.h.
49. **Seriani, S., Cortellessa, A., Belfio, S., Sortino, M., Totis, G., Gallina, P. (2015).** Automatic path-planning algorithm for realistic decorative robotic painting. Automation in Construction, Vol. 56, pp. 67–75. DOI: 10.1016/j.autcon.2015.04.016.
50. **Shih, H. T., Yu, C. L., Wu, M. T., Liu, C. S., Tsai, C. H., Hung, D. Z., Wu, C. S., Kuo, H. W. (2011).** Subclinical abnormalities in workers with continuous low-level toluene exposure. Toxicology and industrial health, Vol. 27, No. 8, pp. 691–699. DOI: 10.1177/0748233710395348.
51. **Singh, P., Choudhary, S. K. (2021).** Introduction: Optimization and metaheuristics algorithms. Springer Singapore, pp. 3–33. DOI: 10.1007/978-981-15-7571-6_1.
52. **Sun, R., Xu, K., Ji, S., Pu, Y., Man, Z., Ji, J., Chen, M., Yin, L., Zhang, J., Pu, Y. (2020).** Benzene exposure induces gut microbiota dysbiosis and metabolic disorder in mice. Science of The Total Environment, Vol. 705, pp. 135879. DOI: 10.1016/j.scitotenv.2019.135879.
53. **Susetyo, S. H., Oginawati, K., Utami, R. R. (2022).** Effect of cadmium exposure on the potential for impaired kidney function on the manual car painting industry. Trends in Sciences, Vol. 19, No. 5, pp. 2882–2882. DOI: 10.48048/tis.2022.2882.
54. **Talbi, E. G. (2009).** Metaheuristics: From design to implementation. Wiley Publishing.
55. **Thirion, V., Gilles, B., Pereira, A., Hascoet, J. Y. (2017).** Robust and efficient scheduling of painting operations in industrial robotics. International Journal of Production Research, Vol. 55, No. 23, pp. 7095–7110. DOI: 10.1080/00207543.2017.1371386.
56. **Valdés, A., Zanobetti, A., Halonen, J. I., Cifuentes, L., Morata, D., Schwartz, J. (2012).** Elemental concentrations of ambient particles and cause specific mortality in Santiago, Chile: A time series study. Environmental Health, Vol. 11, pp. 1–8. DOI: 10.1186/1476-069X-11-82.

57. **Villani, V., Cagnina, L., Bruni, L. E. (2019).** An approach for scheduling collaborative robots using graph coloring and multi-agent systems. IEEE Access, Vol. 7, pp. 47612–47622. DOI: 10.1109/ACCESS.2019.2905594.

58. **Wang, J., Xu, L., Pan, Z. (2018).** Collaborative multi-robot painting system based on industry 4.0. Procedia CIRP, Vol. 72, pp. 1069–1074. DOI: 10.1016/j.procir.2018.03.235.

59. **Wang, L., Zhang, Y., Jin, X., Qiu, Z. (2017).** Collaborative robots in industry 4.0: A survey of the state of the art. IEEE Access, Vol. 5, pp. 4653–4668. DOI: 10.1109/ACCESS.2017.2675252.

60. **Wang, P., Xu, R., Wu, Y. (2019).** Dynamic modeling and simulation of multi-robot system based on Euler-Lagrange equation. Journal of Advanced Mechanical Design, Systems, and Manufacturing, Vol. 13, No. 1, pp. 20180347. DOI: 10.1299/jamdsm.2018jamdsm0347.

61. **Wang, R., Zhang, Y., Lan, Q., Holford, T. R., Leaderer, B., Hoar-Zahm, S., Boyle, P., Dosemeci, M., Rothman, N., Zhu, Y., Qin, Q., Zheng, T. (2009).** Occupational exposure to solvents and risk of non-Hodgkin lymphoma in connecticut women. American Journal of Epidemiology, Vol. 169, No. 2, pp. 176–185. DOI: 10.1093/aje/kwn300.

62. **Weichert, D., Link, P., Stoll, A., Rüping, S., Ihlenfeldt, S., Wrobel, S. (2019).** A review of machine learning for the optimization of production processes. The International Journal of Advanced Manufacturing Technology, Vol. 104, No. 5, pp. 1889–1902. DOI: 10.1007/s00170-019-03988-5.

63. **Wild, C., Weiderpass, E., Stewart, B. W. (2020).** World cancer report: cancer research for cancer prevention. International Agency for Research on Cancer.

64. **Wong, O., Fu, H. (2005).** Exposure to benzene and non-Hodgkin lymphoma, an epidemiologic overview and an ongoing case-control study in Shanghai. Chemico-biological interactions, Vol. 153, pp. 33–41. DOI: 10.1016/j.cbi.2005.03.008.

65. **World Health Organization (2010).** Increasing access to health workers in remote and rural areas through improved retention: Global policy recommendations. World Health Organization.

66. **Yu, H., Yang, W., Huang, J., Zhou, M., Ma, B. (2020).** Task allocation of multiple collaborative robots in smart factory based on improved genetic algorithm. Robotics and Computer-Integrated Manufacturing, Vol. 61, pp. 101837. DOI: 10.1016/j.rcim.2019.101837.

67. **Zbiss, K., Kacem, A., Santillo, M., Mohammadi, A. (2022).** Automatic collision-free trajectory generation for collaborative robotic car-painting. IEEE Access, Vol. 10, pp. 9950–9959. DOI: 10.1109/ACCESS.2022.3144631.

68. **Zhang, J., Wang, H., Yuan, C. (2017).** Controller design for cooperative manipulation of multiple mobile robots based on leader-follower strategy. Journal of Advanced Mechanical Design, Systems, and Manufacturing, Vol. 11, No. 1, pp. 20160260. DOI: 10.1299/jamdsm.2016jamdsm0260.

69. **Zhang, W., Hu, C., Chen, T., Wang, X., Wang, H. (2018).** Adaptive robust control for multi-agent systems with multiple leader-follower architectures: A small gain approach. IEEE Transactions on Cybernetics, Vol. 48, No. 11, pp. 3107–3117. DOI: 10.1109/TCYB.2017.2730199.

70. **Zhang, Y., Tiňo, P., Leonardis, A., Tang, K. (2021).** A survey on neural network interpretability. IEEE Transactions on Emerging Topics in Computational Intelligence, Vol. 5, No. 5, pp. 726–742. DOI: 10.1109/TETCI.2021.3100641.

71. **Zheng, H., Liu, H., Jia, Z., Ma, W., Meng, D., Tan, M. (2017).** Dynamic modeling and adaptive control of collaborative painting robot. Chinese Journal of Mechanical Engineering, Vol. 30, No. 4, pp. 981–989. DOI: 10.3901/CJME.2017.0424.042.

72. Zhu, C. Q., Lam, T. H., Jiang, C. Q. (2001). Lymphocyte DNA damage in bus manufacturing workers. Mutation Research/Genetic Toxicology and Environmental Mutagenesis, Vol. 491, No. 1-2, pp. 173–181. DOI: 10.1016/S1383-5718(01)00141-3.

Article received on 31/01/2024; accepted on 24/04/2024.
**Corresponding author is Lucero Ortiz-Aguilar.*

ISSN 2007-9737

Prescribed-Time Trajectory Tracking Control of Wheeled Mobile Robots Using Neural Networks and Robust Control Techniques

Jesus A. Rodríguez-Arellano[1], Víctor D. Cruz[1], Luis T. Aguilar[1,*], Roger Miranda-Colorado[2,3]

[1] Instituto Politécnico Nacional, Tijuana, Mexico

[2] Instituto Politécnico Nacional, CITEDI, Mexico

[3] Cátedras CONAHCyT, Mexico

jrodriguez@citedi.mx, vcruz@citedi.mx, laguilarb@ipn.mx, rmirandaco@conacyt.mx

Abstract. This research presents a novel trajectory generation algorithm and the design of a prescribed time controller for trajectory tracking tasks for autonomous vehicles. The trajectory generation algorithm uses a hybrid combination of computer vision techniques and intelligent rail detection methods using an on-board camera. Based on the previous information, a possible trajectory is then generated that the vehicle should follow. A time-prescribed controller is then developed and implemented to track the trajectory generated by the proposed methodology. The controller uses a hybrid structure in which a time-varying feedback controller transitions into a fixed-time controller. This approach achieves stabilization in the prescribed time despite the initial conditions. To address the trajectory design, a scaled autonomous vehicle simulator was used to then evaluate the prescribed time controller compared to a finite time controller and a dynamic feedback controller. The simulation results demonstrate the effectiveness of trajectory generation and trajectory tracking control algorithms in addressing these challenges in real-world scenarios by examining two situations: unperturbed and perturbed cases.

Keywords. Prescribed time stabilization, trajectory generation, neural networks.

1 Introduction

Research on autonomous vehicles is intensively explored due to their increasing scope of application, such as surveillance tasks, space exploration, delivery, transportation, and others [1, 2, 3, 4]. These vehicle systems require information about the environment to correctly perform the various tasks and avoid collisions with various obstacles in the environment. The various tasks that these vehicles perform are classified into three main problems: trajectory tracking, path tracking, and point stabilization [5,6].

All information is collected by the sensors these systems are equipped with, such as B. LiDAR, ultrasonic sensors, GPS, and cameras. The information of the environment is processed and interpreted to perform the above-mentioned navigation tasks and avoid accidents and collisions [4, 7]. Therefore, the main focus of this study consists of the precise control of the vehicle's movement along a desired path, known as trajectory tracking, as well as the generation of such paths, known as trajectory generation. These topics have attracted significant attention in the field of autonomous vehicles [8]. A wide range of studies have been carried out on the generation of trajectories.

Prior studies [9, 10, 11, 12] have shown that it is possible to implement intelligent techniques for

generating trajectories and segmenting lanes. A technique for lane line identification is proposed by the authors in [13]. This approach utilizes the RANSAC algorithm to aid in the detection of lane lines and is predicated on an adaptive region of interest extraction strategy.

In addition, convolutional neural network-based algorithms for lane line detection are proposed by Haixia and colleagues [14]. For training and validation purposes, their algorithms employ the TuSimple dataset. The implementation of a Neural Network (NN) using an on-board camera in [9] leads to favorable outcomes in the trajectory generation process.

Hart et al. [10] describe the implementation of intelligent methodologies to create a viable trajectory generator. The study's findings indicate that it is feasible to create trajectories using intelligent methodologies. However, Bellusci et al. [11] present a new approach to lane segmentation using neural networks and computer vision techniques to create a map of the surroundings. Nevertheless, it does not explain the process of generating trajectories.

Besides, in [15] the system utilizes a convolutional neural network along with an auxiliary layer to detect the borders of lanes. In addition, the authors suggest a straightforward algorithm to rectify the vehicle's orientation by utilizing the centroid of the drivable area.

Unfortunately, there is no process for generating trajectories. In addition, in [16] was developed an independent navigation system using machine learning and computer vision techniques on a scaled vehicle.

The system also incorporates a depth camera for localization. However, the algorithm's performance is reduced in low-brightness scenarios. Furthermore, a study conducted by Neven et al. [17] focuses on the development of a control system for a Car-Like robot equipped with a vision system.

This system enables the robot to detect and monitor lanes on a road. The study's findings indicate that the utilization of vision techniques leads to effective lane detection in practical situations.

After addressing the trajectory generation problem, the subsequent critical task is solving trajectory tracking. Numerous studies have tackled this challenge, but there are lingering issues in the field. Various control schemes, including feedback control strategies [18], Sliding Mode Control (SMC) [19], and decoupled approaches [20], have been explored.

Furthermore, these controllers use different scenarios that encompass diverse convergence rates [20, 21], the effect of disturbances [19, 22, 23], and different kinematic models [18]. Cui et al. [24] introduced an adaptive control law within SMC, demonstrating exponential convergence to the trajectory.

The proposed methodology implements a decoupled approach to position and orientation tracking. Experimental results indicated its effectiveness for trajectory tracking despite disturbances. Nevertheless, the proposed methodology implements a simplified kinematic model, and the convergence rate is low. Qun Lu et al [21] proposed a fixed-time controller coupled with an observer under kinematic disturbances.

The controller accounted for signal saturation to prevent slipping, yielding satisfactory tracking results. However, it is noteworthy that the convergence exhibited a gradual pace. The employed kinematic model was the simplified version, and the control structure is complex. In [25], the authors presented a prescribed-time containment controller coupled with a prescribed-time observer to achieve leader-follower tasks.

This study employs the effect of uncertainties and external disturbances by using a chain of integrators for the model. The results demonstrated good tracking performance and disturbance rejection. The research on trajectory tracking has been tackled from different perspectives, like finite-time stability, fixed-time stability, and simplified kinematic models.

However, prescribed-time stability has not been widely studied on WMRs, but in [25] it was proved that this methodology can be implemented in these systems by achieving a fast convergence rate and low tracking errors. Thus, it is important to tackle this problem in WMRs because these systems need to attain a fast response in different scenarios where convergence time is crucial.

Furthermore, trajectory generation is a problem that has been studied from different perspectives, however, there are not many studies with onboard cameras on this crucial task.

1.1 Contribution

Based on the previous literature review, the contribution of this research is to address the trajectory generation and trajectory tracking tasks for autonomous vehicles.

To attain the trajectory generation problem, we develop a novel algorithm that combines computer vision techniques and NNs that enable us to segment rails and then design a feasible trajectory using an onboard camera.

We generate the trajectory using the Autominy simulator and propose a novel methodology. Furthermore, we design a new prescribed-time controller that drives the vehicle to the desired trajectory despite the effects of the disturbances.

This controller is composed of two stages: initially, a time-varying feedback control drives the system to a neighborhood of the origin; then it switches to a twisting controller that converges in fixed time to the origin.

To implement the proposed controller, we perform a coordinate transformation to the complete kinematic model of a Car-Like robot. Then, a series of simulations are performed between the proposed controller against a finite-time controller and a feedback controller. The trajectory tracked is the reference signal generated by the proposed algorithm.

The results demonstrate that the desired trajectory is a good option for trajectory generation problems, and the proposed controller is also a feasible option for trajectory tracking by demonstrating its superior performance against the compared control schemes. Then, the main contributions are:

– Develop a novel trajectory generation algorithm by combining NNs and computer vision techniques for WMRs using an on-board camera.
– Design of a novel prescribed time controller using the complete kinematic model of a WMR, that attains the trajectory tracking problem despite the effect of kinematic disturbances.
– Validation of the trajectory generated by the proposed algorithm by comparing the proposed controller and control schemes from the literature.

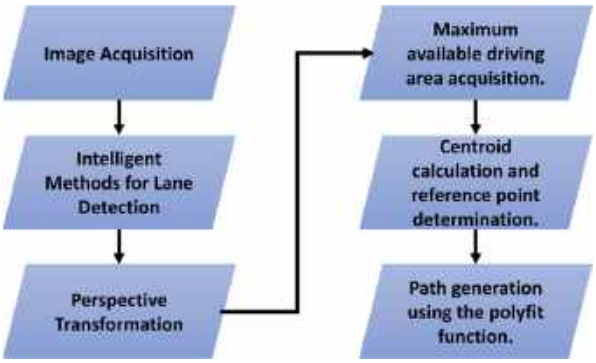


Fig. 1. Trajectory generation methodology

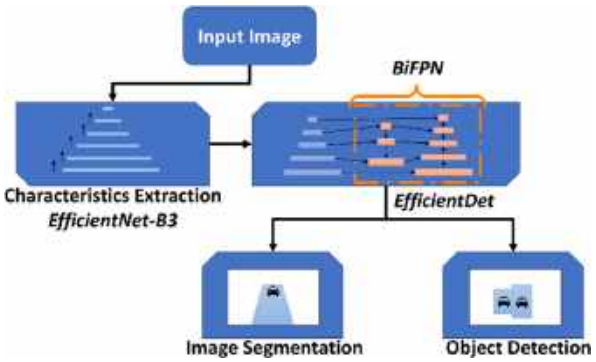


Fig. 2. HybridNets architecture [29]

– Exhaustive qualitative and quantitative study that demonstrates the superiority of the proposed controller against the finite time and dynamic feedback controllers.

1.2 Organization

The subsequent sections are arranged in the following order: Section 2 describes the novel approach for generating trajectories, detailing both the intelligent method and the vision techniques. Section 3 provides a detailed explanation of the kinematic model of a WMR, including a coordinate transformation that allows for the implementation of a hybrid control scheme.

Section 4 develops the controller design that attains the prescribed time stabilization. The outcomes of the trajectory generation methodology are showcased in Section 5, alongside the evaluation of the suggested controller formulated in Section 4. Section 6, ultimately, provides the final findings and results of this manuscript.

Fig. 3. Image acquired from the on-board camera of the Autominy simulator

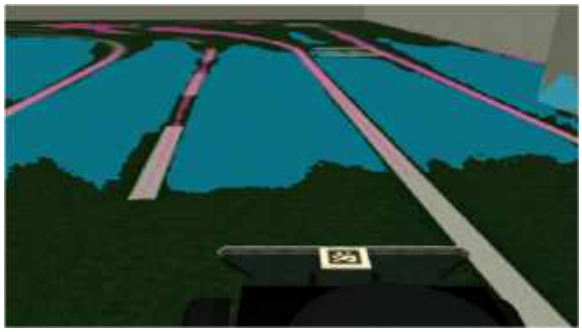


Fig. 4. Output image with HybridNets NN

1.3 Notation

Trigonometric functions are described as c_ψ, s_ψ, t_ψ which corresponds to $\cos(\psi), \sin(\psi)$ and $\tan(\psi)$ correspondingly. The function $\text{sign}(\sigma) = \sigma/|\sigma|$ if $\sigma \neq 0$, and $\text{sign}(0) \in [-1,1]$. $\mathbb{R}_+$ represents positive real numbers.

2 Trajectory Generation

The generation of trajectories is an essential undertaking for WMRs owing to the diverse outcomes it has in real-world situations and the limitations it must satisfy to ensure its tracking is feasible. To achieve this goal, we propose the use of the methodology presented in Fig. 1, which integrates computer vision techniques and neural networks. To achieve this objective, we provide a detailed description of the proposed strategy.

2.1 HybridNets Neural Network

The Neural Network employed is HybridNets1[1] [27, 28], which is an end-to-end perception neural network based on PyTorch. The objective is to address the multi-task issue by employing segmentation and box detection classification networks. Its main architecture comprises two networks, as depicted in Fig. 2.

The initial part of the system is the backbone, which uses the EfficientNet-B3 convolutional neural network architecture to extract characteristics from the input. This architecture scales the dimensions of depth, width, and resolution using a composite coefficient to obtain feature maps of the image.

The information extracted by the backbone is then passed on to the neck network, called EfficientDet, which uses a Weighted Bi-directional Feature Pyramid Network (BiFPN) module for image segmentation and object detection. The BiFPN module achieves this by creating bidirectional interconnections between network nodes. Each input feature is assigned an additional weight, allowing the network to determine the individual significance of each feature[2].

2.2 Proposed Algorithm

To attain the trajectory generation task, we perform a series of steps detailed in Fig. 1, which allows us to finally obtain a feasible trajectory for a WMR using an on-board camera.

To this end, we provide a deep description of this algorithm in this section by using the Gazebo simulator of the Autominy vehicle [31].

Image acquisition. The first step entails reading the input image captured by the vehicle's built-in camera, as shown in Fig. 3.

This step can be realized through the application of the available topics and the Robotic Operating System (ROS) to control the vehicle.

[1] The TuSimple dataset, comprising 6408 images of highways in the United States presented at a resolution of 1280x720, was utilized to train HybridNets [26].

[2] To implement the intelligent method, the NN was converted to an *Open Neural Network Exchange* (ONNX) model to enhance its inference. The codes can be found in [29], while the weights can be found in [30].

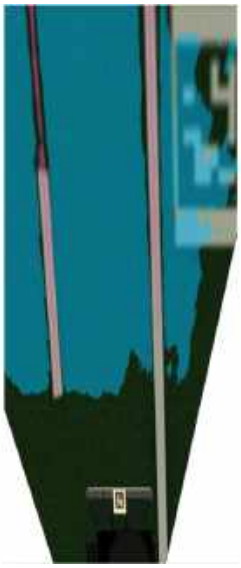


Fig. 5. Bird-Eye-View perspective transformation

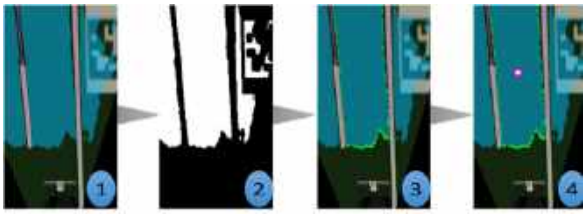


Fig. 6. Procedure for determining the centroid of the available drivable area

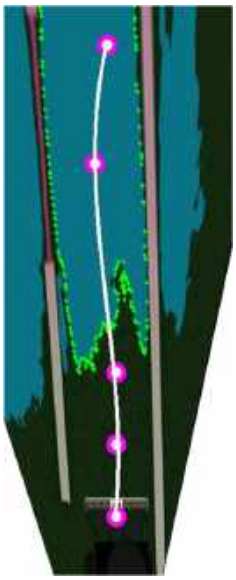


Fig. 7. Reference points for path visualization

Intelligent Methods for Lane Detection. HybridNets NN application is used for lane segmentation and lane detection. This scheme identifies and separates the lane observed by the vehicle's on-board camera. An important advantage of this NN is its ability to accurately identify the lane directly above the vehicle, as shown in Fig. 4.

Perspective Transformation. The output NN image undergoes a perspective transformation technique called "bird's eye view". This technique simulates the image being viewed from a higher angle, similar to a bird's viewpoint [32].

Leveraging this transformation provides a comprehensive perspective of the lane, ensuring both lanes remain parallel to facilitate the implementation of lane detection and segmentation processes. This simplifies the detection and segmentation processes. Figure 5 illustrates the use of the technique from a bird's eye view.

Maximum Drivable Area Acquisition. In our pursuit to determine the largest navigable region, we first examine the area divided by the NN and thus determine the track with the largest extent to generate the desired path. To achieve this, a mask is implemented, which makes use of the segmentation color indicated by the intelligent method results (blue area).

All resulting available segmented areas are then found and the maximum available area is calculated. Subsequent to this phase, the largest area is selected and highlighted in green to proceed with the trajectory generation method.

The algorithm then identifies all currently available segmented areas and uses them to calculate the largest possible area. Next, the algorithm identifies the largest area and marks it with a green marker to proceed with trajectory creation.

Centroid Calculation and Reference Point Determination. The subsequent procedure entails determining the centroid of the drivable area that has been identified through the green contour in the segmented results, as depicted in Fig. 6. Afterward, five points are positioned to create the intended trajectory. Initially, two immobile points are positioned at an equivalent elevation as the ArUco marker on the Autominy vehicle. These

points are fixed and serve as the initial positions for displaying the trajectory. In addition, two extra points are included, one located blow, and one above the calculated centroid. The reference points are adjusted according to the area determined by the NN, and they help to determine the trajectory when navigating a curved lane based on the captured image.

Afterward, the visual representation of the trajectory that corresponds to the five points mentioned earlier is displayed in Fig. 7. Additionally, the homography transformation converts the image representing the aerial perspective into the initial image perspective [33].

This transformation is dependent on the ArUco position, which serves as a reference point. This procedure utilizes the marker location and its position within the pixel coordinates on the bird-eye view image to accurately convert them into the corrected perspective pixel coordinates.

Figure 8 illustrates the process of translating the first trajectory to its equivalent on the vehicle's perspective.

Path generation. Ultimately, we derive the equation that defines the reference trajectory. The outcomes of the trajectory generation phase utilizing our suggested approach are depicted in Fig. 9. The trajectory is determined by analyzing the image captured by the on-board camera.

This visual observation is used to determine the trajectory. Figure 10 illustrates the reference trajectory produced using the proposed methodology in conjunction with the Autominy simulator. The controller will receive this trajectory as the reference input.

3 Kinematic Model and Control Design

To tackle the trajectory tracking problem, we aim to design a controller that achieves prescribed-time stability to the desired trajectory. To this end, let us consider the full kinematic model a Car-Like robot depicted in Fig. 11, where $\boldsymbol{q}(t) = [x(t),\ y(t),\ \theta(t),\ \phi(t)] \in \mathbb{R}^4$ is the system's vector configuration, the $x(t), y(t)$ is the position on the plane with respect to the world frame $\{x, y\}$, $\theta(t)$ is the orientation of the vehicle with respect to the x axis, $\phi(t)$ is the

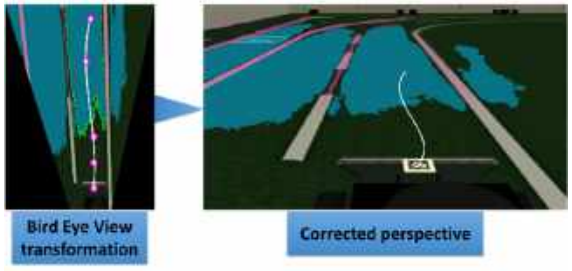


Fig. 8. Perspective correction

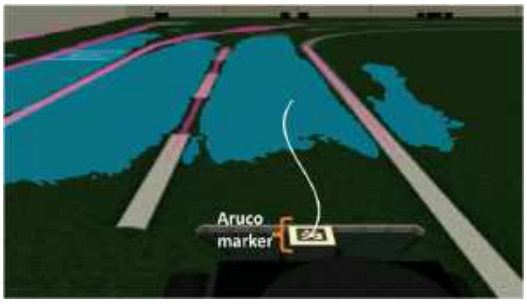


Fig. 9. Results for trajectory generation

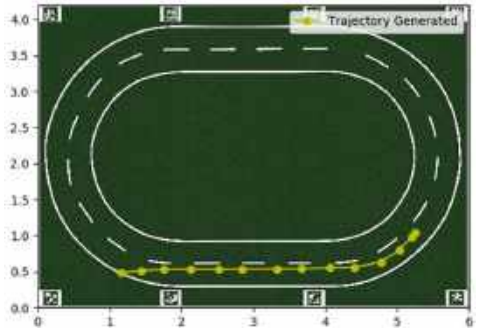


Fig. 10. Path generated using the proposed trajectory generation method

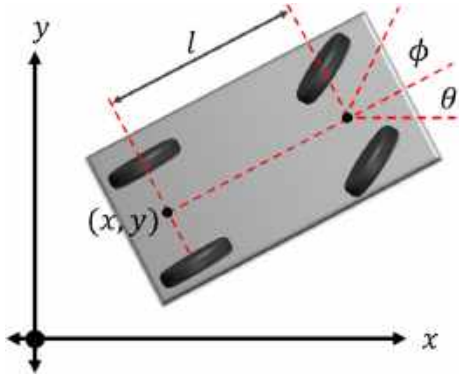


Fig. 11. Description of the WMR's kinematic model in the x, y plane

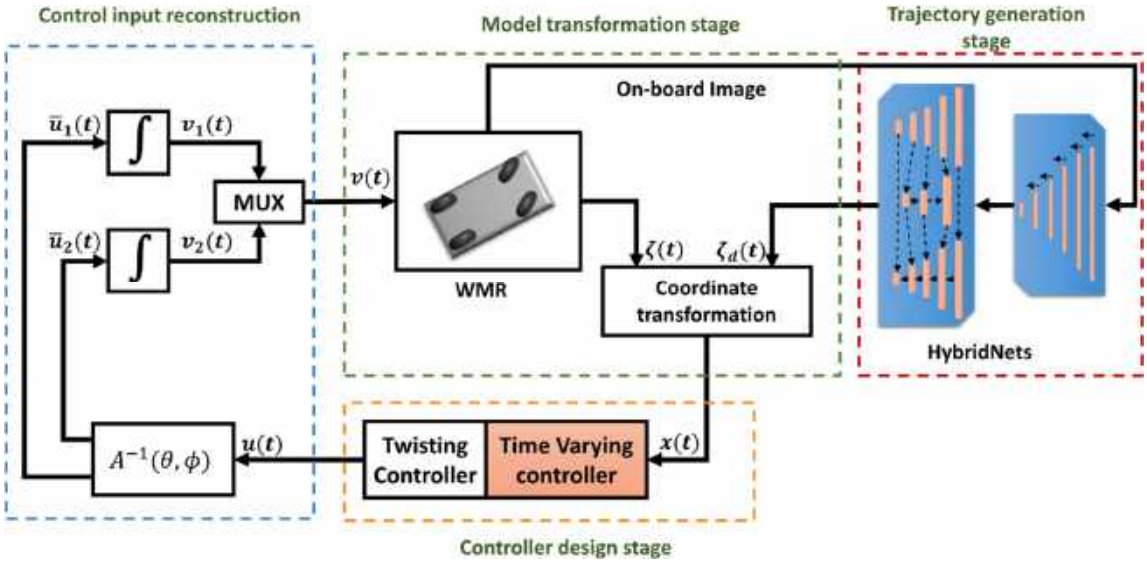


Fig. 12. General scheme of the proposed methodology

steering angle of the front wheels. Now, the kinematic model of the WMR is described as:

$$\dot{\boldsymbol{q}}(t) = S(q)\boldsymbol{v}(t) + \boldsymbol{d}(t), \tag{1}$$

$$S(\boldsymbol{q}) = \begin{bmatrix} c_\theta & s_\theta & \frac{t_\phi}{l} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix}, \tag{2}$$

With $v(t) = [v_1(t), v_2(t)]^T \in \mathbb{R}^2$ is the control input vector with $v_1(t), v_2(t)$ being the linear and angular velocities, and $\boldsymbol{d}(t) = [d_1(t),\ d_2(t),\ d_3(t),\ d_4(t)]^T \in \mathbb{R}^4$ encompasses the disturbances, which are bounded and smooth until its first time derivative [34].

Furthermore, the actuators generate bounded control signals. Hence, based on the previous statement we are able to consider the following assumption.

Assumption 1. There exist some positive constants $\bar{\mathrm{D}}, \bar{\mathrm{d}}, \mathrm{V}_1^+, \mathrm{V}_2^+$, and Φ such that:

$$\begin{aligned} &\left|\left|\boldsymbol{d}(t)\right|\right| \leq \bar{D}, \left|\left|\dot{\boldsymbol{d}}(t)\right|\right| \leq \bar{d}, |v_1(t)| \leq V_1^+, \\ &|v_2(t)| \leq V_2^+, |\phi(t)| \leq \Phi < \pi/2. \end{aligned} \tag{3}$$

Then, we perform a coordinate transformation by defining the new output variable:

$$\zeta(\mathrm{t}) = \begin{bmatrix} \zeta_1(t) \\ \zeta_2(t) \end{bmatrix} = \begin{bmatrix} x + l\, c_\theta + \delta c_{(\theta+\phi)} \\ y + l\, s_\theta + \delta\, s_{(\theta+\phi)} \end{bmatrix}. \tag{4}$$

With an arbitrary $\delta \neq 0$, which will be used to design a controller. Furthermore, to attain the trajectory tracking problem, a reference kinematic model is required, which is described by:

$$\begin{aligned} \dot{\boldsymbol{q}}_{\boldsymbol{d}}(t) &= S(q_d)\boldsymbol{v}_{\boldsymbol{d}}(t), \\ S(\boldsymbol{q}_{\boldsymbol{d}}) &= \begin{bmatrix} c_{\theta_d} & s_{\theta_d} & \frac{t_{\phi_d}}{l} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\ \boldsymbol{v}_{\boldsymbol{d}}(t) &= \begin{bmatrix} v_{d1}(t) \\ v_{d2}(t) \end{bmatrix}, \end{aligned} \tag{5}$$

where $\boldsymbol{q}_{\boldsymbol{d}}(t) = [x_d(t),\ y_d(t),\ \theta_d(t),\ {}_d(t)]^T \in \mathbb{R}^4$, $v_{d1}(t), v_{d2}(t) \in \mathbb{R}$ are the reference signals of $\boldsymbol{q}(t)$ and $v_1(t), v_2(t)$. Moreover, we perform the same coordinate transformation as in (4):

$$\boldsymbol{\zeta}_{\boldsymbol{d}}(\mathrm{t}) = \begin{bmatrix} \zeta_{d1}(t) \\ \zeta_{d2}(t) \end{bmatrix} = \begin{bmatrix} x_d + l\, c_{\theta_d} + \delta c_{(\theta_d+\phi_d)} \\ y_d + l\, s_{\theta_d} + \delta\, s_{(\theta_d+\phi_d)} \end{bmatrix}. \tag{6}$$

By following the procedure described in [20] along the coordinate transformation (5) and the reference model (9), we define the tracking error $\tilde{\boldsymbol{\xi}}(t) = \boldsymbol{\xi}(t) - \boldsymbol{\xi}_{\boldsymbol{d}}(t)$, which yields to the following dynamic error structure:

$$\begin{aligned}\ddot{\tilde{\boldsymbol{\xi}}}(t) = A(\theta,\phi)\overline{\boldsymbol{u}}(t) + \bar{A}(\theta,\phi)\boldsymbol{v}(t) + \dot{\boldsymbol{\Gamma}}(\boldsymbol{t}) \\ - A(\theta_d,\phi_d)\overline{\boldsymbol{u}}_{\boldsymbol{d}}(t) \\ - \dot{A}(\theta_d,\phi_d)\boldsymbol{v}_{\boldsymbol{d}},\end{aligned} \tag{7}$$

where the structure of $A(\theta,\phi)$, $\bar{A}(\theta,\phi)$, $\overline{\boldsymbol{u}}(t)$, $\dot{\boldsymbol{\Gamma}}(t)$, $\overline{\boldsymbol{u}}_{\boldsymbol{d}}(t)$, $\dot{A}(\theta_d,\phi_d)$, and $\boldsymbol{v}_{\boldsymbol{d}}(t)$ can be consulted in [23].

Now we can state the control objective by implementing a hybrid control scheme that makes the tracking error signal $\tilde{\boldsymbol{\xi}}(t)$ to converge to zero in a prescribed time despite disturbances.

Control objective. Considering the WMR's kinematic model (1), the coordinate transformation (4), and the error's dynamics (7), design a control input $\boldsymbol{v}(t)$ such that the tracking error converges to zero in a prescribed time.

4 Control Design

In order to design the proposed controller, depicted in Fig. 12, we begin by presenting a general structure for the controller, which is then applied to the kinematic model (1) by employing the tracking error $\tilde{\boldsymbol{\xi}}(t)$, and the error's dynamics (7) to achieve prescribed time stability to the desired trajectory.

4.1 General Structure

The control problem encompasses two stages. First, the system is directed towards an arbitrary small attraction zone by means of a time-varying state feedback control law.

Once the system enters the attraction zone, a twisting controller is executed to ensure that it converges to the equilibrium point in prescribed time [35].

In order to address this issue, we begin by formulating a general second-order system with the following structure:

$$\begin{aligned}\dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= u(t,x) + w(t,x),\end{aligned} \tag{8}$$

where $\boldsymbol{x}(t) = [x_1, x_2]^T \in \mathbb{R}^2$ is the state vector, $u(t,\boldsymbol{x}) \in \mathbb{R}$ is the control input, and $w(t,\ \boldsymbol{x}) \in \mathbb{R}$ encompasses smooth and bounded disturbances, such that $|w(t,\ x)| \leq\ W^+$, with $W^+ \in \mathbb{R}$ being the upper bound of the disturbances. To attain prescribed time stability, we structure the control input as follows:

$$u = \begin{cases} u_1(t,\boldsymbol{x}), & t < T_1 \text{ and } \boldsymbol{x} \in \mathbb{R}^2 \setminus G_R, \\ u_2(\boldsymbol{x}), & \text{otherwise} \end{cases}, \tag{9}$$

where $G_R = \{\boldsymbol{x}: V(\boldsymbol{x}) \leq R\}$ is the attraction domain, and:

$$u_1(t,x) = -l_1(t)x_1(t) - l_2(t)x_2(t), \tag{10}$$

$$u_2(x) = -c_1\text{sign}(x_1) - c_2\text{sign}(x_2), \tag{11}$$

$$\begin{aligned}l_1(\mathrm{t}) = \mu^2(\mathrm{t})[2+m][3+m] + k_1\mu(\mathrm{t})[2+m] \\ + k_2\mu^{3+m}(\mathrm{t})[2+m] \\ + k_1k_2\mu^{2+m}(\mathrm{t}),\end{aligned} \tag{12}$$

$$l_2(\mathrm{t}) = 2\mu(\mathrm{t})[2+m] + k_1 + k_2\mu(\mathrm{t})^{2+m}.$$

With $m \in \mathbb{N}$, $k_1, k_2 \in \mathbb{R}_+$, and the function:

$$\mu(t) = \frac{1}{T_1 + t_0 - t}. \tag{13}$$

The control signal $u_1(t,\boldsymbol{x})$ employs some time-varying gains $l_1(t)$ and $l_2(t)$, which act in the first stage of the control strategy in $t < T$. Then the twisting controller, represented by $\mathrm{u}_2(\mathrm{t})$ with some gains $c_1 > c_2 > W^+ > 0$, is introduced once the trajectories of the system enter the attraction domain G_R, defined by the parameter R and the time T. Furthermore, this domain is set by the non-strict Lyapunov function:

$$V(\boldsymbol{x}) = c_1|x_1(t)| + \frac{1}{2}x_2^2(t). \tag{14}$$

Which ensures that the equilibrium point is finite-time stable, according to [36, Theorem 5.1]. The proposed controller attains prescribed-time stabilization by combining two methodologies; the first stage consists in a control structure encompassed by time-varying gains (12) that drives the trajectories of the system to a vicinity of the origin defined by the time T and the attraction domain G_R.

Once the trajectories reach this vicinity, the twisting controller (11) is introduced to reach the origin in fixed time with an admissible disturbance (for more details, refer [35]). Based on the previous information, the controller's implementation for the WMR is described in the following subsection.

4.2 Wheeled Mobile Robot Controller

To implement the hybrid control scheme (9), we first define the following state variables:

$$x_1(t) = \tilde{\xi}_1(t),\ x_2(t) = \dot{\tilde{\xi}}_1(t), \tag{15}$$

$$x_3(t) = \tilde{\xi}_2(t),\ x_4(t) = \dot{\tilde{\xi}}_2(t). \tag{16}$$

Thus, we can rewrite the error's dynamics (7) by using definitions (15, 16), which yields to two decoupled second-order systems:

$$\Upsilon_1 \begin{cases} \dot{x}_1(t) = x_2(t), \\ \dot{x}_2(t) = u_1(t,x) + w_1(t,x), \end{cases} \tag{17}$$

$$\Upsilon_2 \begin{cases} \dot{x}_3(t) = x_4(t), \\ \dot{x}_4(t) = u_2(t,x) + w_2(t,x), \end{cases} \tag{18}$$

where $u_1(t,\boldsymbol{x})$ and $u_2(\boldsymbol{x})$ are the control inputs retrieved from:

$$\boldsymbol{u}(\mathrm{t}) = A(\theta,\phi)\bar{\boldsymbol{u}}(\mathrm{t})\,. \tag{19}$$

Being:

$$\boldsymbol{u} = [u_1(t,x), u_2(t,x)]^T,$$

$$\begin{aligned} w_1 = \dot{a}_{11}v_1 + \dot{a}_{12}v_2 + \dot{\lambda}_1 - a_{d11}\dot{v}_{d1} - a_{d12}\dot{v_{d2}} \\ - \dot{a}_{d11}v_{d1} - \dot{a}_{d12}v_{d2}, \end{aligned} \tag{20}$$

$$\begin{aligned} w_2 = \dot{a}_{21}v_1 + \dot{a}_{22}v_2 + \dot{\lambda}_2 - a_{d21}\dot{v}_{d1} - a_{d22}\dot{v_{d2}} \\ - \dot{a}_{d21}v_{d1} - \dot{a}_{d22}v_{d2}, \end{aligned}$$

where $w_i(t)$ are considered smooth and bounded disturbances, such as $|w_i(x,t)| \leq M_i^+$ with $i \in \{1,2\}$. According to (9), the hybrid controller is structured as [35]:

$$u_i = \begin{cases} u_{1i}, & t < T_{1i} \text{ and } \boldsymbol{x} \in \mathbb{R}^2 \setminus G_{Ri} \\ u_{2i}, & \text{otherwise,} \end{cases} \tag{21}$$

where $T_{1i} > 0$ is a design parameter, and:

$$u_{11} - l_{11}(t)x_1(t) - l_{12}x_2(t), \tag{22}$$

$$u_{12} = -l_{21}(t)x_3(t) - l_{22}x_4(t), \tag{23}$$

$$u_{21} = -c_{11}\mathrm{sign}(x_1) - c_{12}\mathrm{sign}(x_2), \tag{24}$$

$$u_{22} = -c_{21}\mathrm{sign}(x_3) - c_{22}\mathrm{sign}(x_4), \tag{25}$$

$$G_{R1} = \{\boldsymbol{x}: V(x_1,x_2) \leq R_1\}, \tag{26}$$

$$G_{R2} = \{\boldsymbol{x}: V(x_3,x_4) \leq R_2\}. \tag{27}$$

The control $u_{1i}(t,x)$ is a linear time-varying state feedback with positive time-varying gains $l_{1i}(t),\ l_{2i}(t)$ with the structure defined in (12), and the control structure $u_{2i}(x)$ attains stability in finite time to the origin according to the stability analysis developed with the non-strict Lyapunov function (14) [36].

Finally, in order to recover the control inputs $v_1(t)$, and $v_2(t)$ we use the definitions $A(\theta,\phi)$ and $\dot{\boldsymbol{v}}(t)$ from (19), which yields to:

$$\boldsymbol{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix} = \begin{bmatrix} \int_0^{\tau} \dot{v_1}(\tau)\,d\tau \\ \int_0^{\tau} \dot{v_2}(\tau)d\tau \end{bmatrix}, \tag{28}$$

where $\bar{\boldsymbol{u}}(t) = A^{-1}(\theta,\phi)\boldsymbol{u}(t)$. Therefore, we synthesize the stated controller in the following Theorem.

Theorem 1. [35] Let the dynamic's error (7) and assume that **A1** holds. Then, the controller (19), (21)-(28), provided (12, 13), ensures the trajectory tracking in prescribed time.

5 Numerical Results

To assess the proposed approaches for trajectory generation and trajectory tracking tasks, we conducted simulations using MATLAB/SIMULINK® for trajectory tracking and the Autominy simulator for trajectory generation [4]. The trajectory generated via the methodology described in Section 2 is illustrated in Fig. 13.

To assess the performance of the proposed methodology, named **PTC**, we consider the controllers described in references [18] and [20]. We choose the dynamic feedback controller [18] (named **DFC**), due its simple structure and because it implements the complete kinematic model of a Car-Like robot.

Furthermore, we also employ [20] (referred to as **FTC**) because it utilizes the full kinematic model of the Car-Like robot, attains the trajectory tracking problem by using a decoupling approach, and achieves Finite-time stability. The following cases are considered for the validation of the PTC controller:

– **C1**: There is no effect of kinematic disturbances in the model, considering the

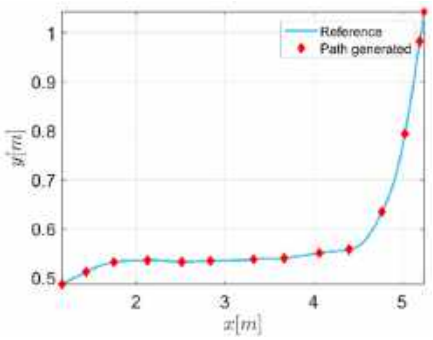


Fig. 13. Reference points for path visualization

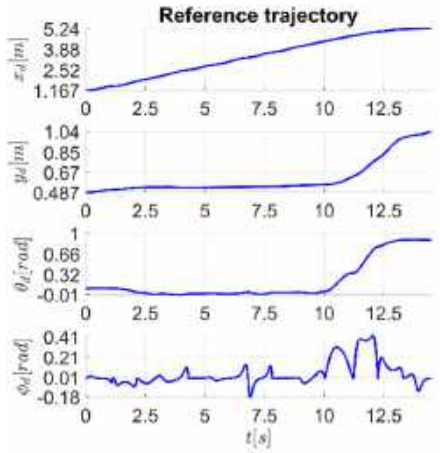


Fig. 14. Desired trajectory represented in generalized coordinates $x(t)$, $y(t)$, $\theta(t)$ and $\phi(t)$

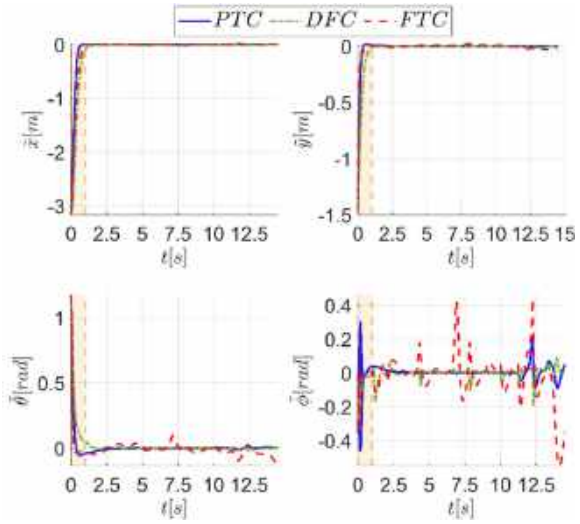


Fig. 15. Tracking errors in $\tilde{x}, \tilde{y}, \tilde{\theta}$ and $\tilde{\phi}$ for case **C1**

initial conditions $[x(0),\ y(0),\ \theta(0),\ \phi(0)]^T = [-1,\ -2,\ \arctan(\pi/2),0]^T$ and $l = 0.255$ [m].

– **C2**: A The effect of kinematic disturbances is introduced with $d_1(t) = 0.05 + 0.05\sin(2t)$, $d_2(t) = -0.05 - 0.05\cos(2t)$, $d_3(t) = 0.05$, $d_4(t) = -0.05$, with the initial conditions considered in case **C1**.

The gains considered for the DFC are $k_{pi} = 343, k_{vi} = 147, k_{ai} = 21$, and for the FTC are $k_1 = 35,\ k_2 = 25,\ k_3 = 5$ and $k_4 = 7$. Finally, the parameters for the PTC's time-varying gains are $T_{1i} = 1,\ \ k_{11} = 5,\ \ k_{12} = 5,\ \ k_{21} = 25,\ \ k_{22} = 15,$ $m\_i = 1$, and for the twisting controller are $c_{11} = 275,\ c_{12} = 495,\ c_{21} = 770,\ c_{22} = 715$ with $\delta = 0.03$. The simulation in Matlab/Simulink was performed using the sampling time of 1×10^{-4} seconds and Runge-Kutta algorithm as the solver.

In Fig. 13 is depicted the final trajectory generated with the procedure detailed in Section 2. The red marks are the samples gathered from the Autominy simulator, and the smooth trajectory generated with the time intervals and the red marks is represented by the blue line. The equations that represent the motion at each time interval are shown in Table 3 (see Appendix A) and are depicted in generalized coordinates in Fig. 14.

In Fig. 15 are depicted the tracking errors in generalized coordinates. It can be observed that the FTC and the PTC controllers converge faster than the DFC. Also, the proposed controller reaches the vicinity near the origin in $\mathrm{t} < 1[\mathrm{s}]$, as highlighted with the orange area; during this transitory stage the time-varying feedback control scheme is used, which afterwards is switched to the twisting controller.

This behavior is also present in $\tilde{\mathrm{y}}$, where the FTC and the PTC present the fastest responses, followed by the DFC. The PTC controller exhibits a slight overshoot; however, it approaches the vicinity of the origin at $\mathrm{t} < 1[\mathrm{s}]$ within the orange region.

Finally, in $\tilde{\phi}$ the *PTC* controller obtains the fastest convergence compared with the DFC and FTC controllers; however, it also exhibits the highest level of overshoot. Furthermore, in $10 \leq t \leq 14.15$ the proposed controller achieves the lowest overshoots compared with the FTC and DFC controllers; this behavior can be attributed

to the alterations in the reference signal for $\phi(t)$ as depicted in Fig.14 during this specific period.

Furthermore, in Fig.16 are depicted the control signals generated by the controllers in comparison to the desired control inputs $v_1(t)$ and $v_2(t)$, which are represented by the black-*colored* dashed lines. In $v_1(t)$, it is evident that the FTC controller exhibits the highest overshoot when compared to the PTC and DFC controllers; this behavior is associated with the fastest convergence of the FTC's tracking $\tilde{x}$ in Fig. 15, which is attributed to the presence of the SMC in the x coordinate.

However, the PTC achieves the fastest response, which implies the fastest tracking response. For $v_2(t)$, the PTC controller generates a large control signal before the commutation of the twisting controller, then it generates a noisy control signal, which is directly related to the tracking error of $\phi(t)$ in Fig. 15. Moreover, it is evident that the PTC controller transitions to the twisting control scheme in a time period less than 1 second.

As observed in the signal, the PTC controller generates the effect of chattering due to the presence of the twisting controller. In addition, all the control signals remain with similar behaviors after the transient stage.

Figure 17 depicts the tracking errors for case **C2** in the x, y, θ and ϕ coordinates. It can be observed that $\tilde{x}$ behaves similarly, with the FTC and PTC controllers achieving the fastest response, even when disturbances are considered. Regarding $\tilde{y}(t)$, it is evident that the FTC and PTC controllers also achieve the fastest response; nevertheless, the performance of the FTC and DFC controllers is diminished due to the generation of greater oscillations.

Furthermore, it is noteworthy that the time-varying stage of the PTC commutes to the twisting controller before the orange-colored area ends, where the twisting control signal is introduced, and thereafter the error signal keeps a neighborhood in zero.

For $\tilde{\theta}$, the fastest response is achieved by the PTC and FTC control schemes, but the PTC controller keep less oscillations. It can also be observed that the DFC controller generates a greater error at the end of the simulation.

For $\tilde{\phi}(t)$, the largest overshoot was generated by the PTC due to the time-varying stage and the

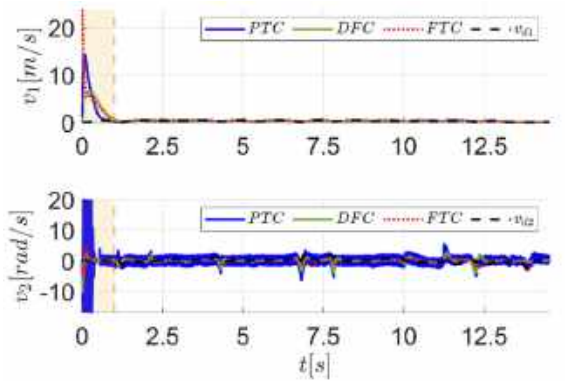


Fig. 16. Control signals $v_1(t)$ and $v_2(t)$ generated for case **C1**

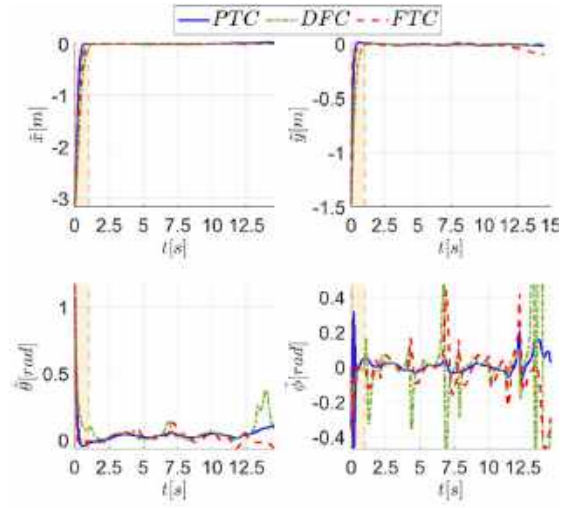


Fig. 17. Tracking errors in $\tilde{x}, \tilde{y}, \tilde{\theta}$ and $\tilde{\phi}$ for case **C2**

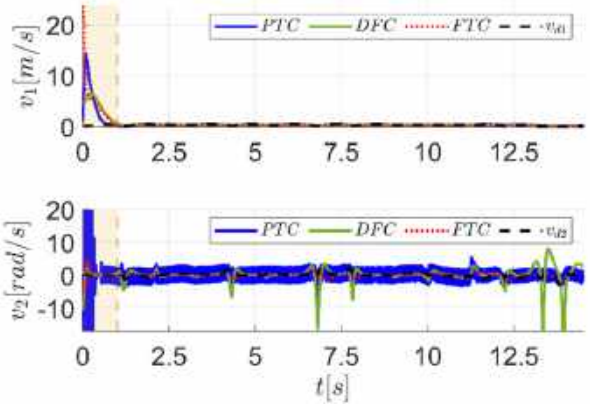


Fig. 18. Control signals $v_1(t)$ and $v_2(t)$ generated for case **C2**

Table 1. Performance indexes IAE, ITSE and ISV for the WMR for case **C1**

Controller	IAE				ISV
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.653**	**0.168**	**0.21**	**0.245**	149.2
DFC	1.415	0.392	0.314	0.299	**28.44**
FTC	0.939	0.26	0.522	1.18	45.03
	ITSE				
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.119**	**0.006**	**0.008**	**0.148**	
DFC	0.29	0.038	0.019	0.111	
FTC	0.264	0.033	0.207	3.242	

Table 2. Performance indexes IAE, ITSE and ISV for the WMR for case **C2**

Controller	IAE				ISV
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.697**	**0.268**	0.62	**0.455**	190.1
DFC	1.373	0.447	1.01	2.03	169.5
FTC	0.936	0.371	**0.544**	1.147	**44.47**
	ITSE				
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.133**	**0.017**	0.253	**0.312**	
DFC	0.49	0.04	1.35	16.26	
FTC	0.263	0.171	**0.159**	2.59	

control signal $v_2(t)$, which is observed at the beginning of the orange-colored area.

The PTC maintains the lowest tracking errors in this coordinate with the lowest oscillations even in the presence of disturbances.

Conversely, the DFC and FTC controllers present the large oscillations during the trajectory tracking. The tracking error $\tilde{\phi}(t)$ is associated with the control signal $v_2(t)$ depicted in Fig. 18, where it is corroborated that the DFC controller generated large control signals. On the other hand, the PTC controller generated the largest overshoot while the time-varying gains were active, i.e., in the orange-colored area at $t < 1[s]$; afterwards, the proposed scheme presents the effect of chattering due to the properties of the twisting controller and the SMC, however, its magnitude is small.

For $v_1(t)$, the PTC controller achieved the fastest response with a slight overshoot compared with the FTC controller. The DFC controller generated the lowest overshot and achieved the lowest response. In addition, it is clear that the proposed PTC controller achieves the fastest response and keeps the lowest oscillations in comparison with the DFC and FTC controllers.

Hence, the aforementioned findings illustrate that the PTC controller achieves the fastest response and the lowest oscillations, despite the existence of disturbances. Therefore, it can be inferred that the PTC controller excels over the DFC and FTC controllers.

In addition, a quantitative analysis was conducted to supplement the qualitative analysis. Tables 1 and 2 expose the IAE (Integral of Absolute Error) and ITSE (Integral of Time-weighted Squared Error) for evaluating the tracking error of the controllers, while the ISV (Integral of Squared Control Signal) is used to quantify the control signals [20]. Table 1 displays the quantitative tracking errors and ISV for case **C1** while Table 2 showcases the performance indexes for case **C2**.

Regarding case **C1**, it is evident that the PTC controller achieves the lowest tracking errors for each coordinate, except for $\phi(t)$, where the FTC controller achieves the lowest tracking errors in terms of ITSE. These quantities indicate that the PTC controller outstands with the best performance by achieving the lowest tracking errors with respect to the DFC and FTC controllers.

In addition, for case **C2**, the FTC controller demonstrated the smallest tracking error in $\theta(t)$, while the PTC controller exhibited the smallest tracking errors in the remaining coordinates, thus demonstrating its superior performance in comparison with the other controllers.

Based on the preceding qualitative and quantitative analysis, we can deduce that the PTC controller demonstrates superior performance in both disturbed and undisturbed scenarios by achieving the fastest response and lowest oscillations with respect to the DFC and PTC controllers. Moreover, the proposed control strategy achieved the lowest tracking errors with ITSE and IAE in general, thus outstanding its performance. Furthermore, we have observed that the trajectory generation algorithm is an effective solution for the task of generating trajectories.

This is because it successfully fulfills the non-holonomic constraints of WMRs. The simulations conducted with the controllers have demonstrated the algorithm's ability to track trajectories in real-world scenarios, even in the presence of disturbances.

Table 3. Polynomials for trajectory reference

Time interval	Reference trajectories
$0 \leq t < 1.13$	$x_d(t) = 0.0776t^7 + 0.4971t^6 - 1.7644t^5 + 1.3442t^4 + 0.1t + 1.668$ $y_d(t) = 0.0074t^7 + 0.0471t^6 - 0.167t^5 + 0.1274t^4 + 9e^{-3}t + 0.4869$
$1.13 \leq t < 2.14$	$x_d(t) = 0.004t^7 + 0.88t^6 - 8.37t^5 + 31.73t^4 - 62.0045t^3 + 66.23t^2 - 36.73t + 9.68$ $y_d(t) = 5e^{-4}t^7 + 752e^{-4}t^6 - 707.7e^{-3}t^5 + 2.638t^4 - 5.069t^3 + 5.328t^2 - 2.909t + 1.1545$
$2.14 \leq t < 3.23$	$x_d(t) = -0.01t^6 + 0.1552t^5 - 0.97t^4 + 3.187t^3 - 5.806t^2 + 5.92t - 1.2013$ $y_d(t) = -15.8e^{-3}t^6 + 246e^{-3}t^5 - 1.569t^4 + 5.262t^3 - 9.797t^2 + 9.625t - 3.38$
$3.21 \leq t < 4.32$	$x_d(t) = 2e^{-4}t^7 + 0.175t^6 - 3.68t^5 + 31.51t^4 - 141.87t^3 + 353.27t^2 - 464.53t + 251.09$ $y_d(t) = -15.2e^{-3}t^6 + 0.327t^5 - 2.91t^4 + 13.672t^3 - 35.83t^2 + 49.682t - 27.951$
$4.32 \leq t < 5.31$	$x_d(t) = 0.5t^6 + 13.8t^5 + 156.6t^4 - 942.8t^3 + 3172.7t^2 - 5659.4t + 4181.8$ $y_d(t) = 36e^{-4}t^6 - 0.1t^5 + 1.161t^4 - 7.117t^3 + 24.431t^2 - 44.536t + 34.212$
$5.31 \leq t < 6.81$	$x_d(t) = 0.1t^6 - 2.9t^5 + 41.1t^4 - 308.7t^3 + 1294.2t^2 - 2870.9t + 2635.1$ $y_d(t) = -4e^{-4}t^6 + 12.2e^{-3}t^5 - 0.1618t^4 + 1.1245t^3 - 4.292t^2 + 8.483t - 6.1832$
$6.81 \leq t < 7.83$	$x_d(t) = -19t^5 + 324t^4 - 2994t^3 + 15515t^2 - 42699t + 48755$ $y_d(t) = 19t^5 + 324t^4 - 2994t^3 + 15515t^2 - 42699t + 48755$
$7.83 \leq t < 9$	$x_d(t) = -8t^5 + 152t^4 - 1638t^3 + 9883t^2 - 31714t + 42282$ $y_d(t) = -0.2t^5 + 4.4t^4 - 48.1t^3 + 292.1t^2 - 944.8t + 1270.5$
$9 \leq t < 10.03$	$x_d(t) = -3t^5 + 57t^4 - 672t^3 + 4435t^2 - 15499t + 22397$ $y_d(t) = 0.2t^5 - 4.1t^4 + 47.1t^3 - 296.7t^2 + 980.6t - 1321$
$10.03 \leq t < 11.25$	$x_d(t) = -0.8t^5 + 19.1t^4 - 242.5t^3 + 1712.4t^2 - 6342.8t + 9588.1$ $y_d(t) = -2t^5 + 64t^4 - 896t^3 + 7053t^2 - 29575t + 51615$
$11.25 \leq t < 12.26$	$x_d(t) = -80t^4 + 1460t^3 - 14040t^2 + 71590t - 150960$ $y_d(t) = -10t^5 + 210t^4 - 3150t^3 + 26780t^2 - 121230t + 228190$
$12.26 \leq t < 13.34$	$x_d(t) = 3t^4 + 79t^3 - 2077t^2 + 17320t - 51070$ $y_d(t) = -2t^5 + 64t^4 - 915t^3 + 7138t^2 - 28147t + 42243$
$13.34 \leq t < 14.51$	$x_d(t) = t^5 - 21t^4 + 344t^3 - 3102t^2 + 14569t - 27559$ $y_d(t) = x_d(t) = t^5 - 21t^4 + 344t^3 - 3102t^2 + 14569t - 27559 + 18543t - 35323$

6 Trajectory Generation

The trajectory generation algorithm obtained specific points from the rail detections, which were subsequently analyzed to calculate two polynomials in the x, y plane. In order to achieve a feasible trajectory for a Car-Like robot [17], it is necessary for these polynomials to be both smooth and continuous. The procedure for designing the equations is described in [37], resulting in the equation's trims presented in Table 3.

7 Conclusions

The manuscript introduced an innovative approach for generating trajectories in vehicular systems by utilizing an on-board camera, computer vision techniques, and intelligent algorithms.

The proposed methodology was evaluated by employing three controllers that successfully achieved the trajectory tracking task. In addition, a prescribed-time controller was introduced for the purpose of trajectory tracking tasks. The

performance of this controller was evaluated by comparing it to two controllers previously discussed in the literature.

Employing the simulation capabilities of MATLAB/SIMULINK, the prescribed-time controller demonstrated be superior, showcasing its resilience and adaptability in maintaining the desired trajectory, irrespective of the presence of disturbances.

The proposed methodology demonstrated its key feature of adjusting the convergence rate and its ability to withstand disturbances. Therefore, this study introduced a novel approach that integrates two approaches to tackle significant challenges in vehicular systems: the prescribed-time controller and the trajectory generation algorithm.

Future research ought to emphasize on enhancing the prescribed time controller by means of reducing or modifying the control structure to mitigate chattering and minimize the overshoot of the control signal of $v_2(t)$.

Furthermore, the trajectory generation algorithm can be enhanced by utilizing optimization algorithms to enhance its desired control signals in relation to generalized coordinates.

Acknowledgments

This work was supported in part by Instituto Politécnico Nacional under Grant SIP 2024-0018 and the program “Investigadoras e Investigadores por México”, Cátedras CONAHCYT, Project No. 537.

References

1. **Zhang, J., Li, S., Meng, H., Li, Z., Sun, Z. (2023).** Variable gain based composite trajectory tracking control for 4-wheel skid-steering mobile robots with unknown disturbances. Control Engineering Practice, Vol. 132, pp. 105428. DOI: 10.1016/j.coneng prac.2022.105428.
2. **Li, L., Cao, W., Yang, H., Geng, Q. (2022).** Trajectory tracking control for a wheel mobile robot on rough and uneven ground. Mechatronics, Vol. 83, pp. 102741. DOI: 10.10 16/j.mechatronics.2022.102741.
3. **Gao, H., Chen, C., Ding, L., Li, W., Yu, H., Xia, K., Liu, Z. (2017).** Tracking control of WMRS on loose soil based on mixed H2/H∞ control with longitudinal slip ratio estimation. Acta Astronautica, Vol. 140, pp. 49–58. DOI: 10.1016/j.actaastro.2017.07.037.
4. **Olayode, I. O., Du, B., Severino, A., Campisi, T., Alex, F. J. (2023).** Systematic literature review on the applications, impacts, and public perceptions of autonomous vehicles in road transportation system. Journal of Traffic and Transportation Engineering (English Edition), Vol. 10, No. 6, pp. 1037–1060. DOI: 10.1016/ j.jtte.2023.07.006.
5. **Wang, C. (2011).** A novel variable structure theory applied in design for wheeled mobile robots. Artificial Life and Robotics, Vol. 16, No. 3, pp. 378–382. DOI: 10.1007/s10015-011-0955-3.
6. **Yan, K., Ma, B. (2023).** Global posture stabilization for the kinematic model of a rear-axle driven car-like mobile robot considering obstacle avoidance. IEEE Robotics and Automation Letters, Vol. 8, No. 9, pp. 5568–5575. DOI: 10.1109/lra.2023.3296351.
7. **Kocsis, M., Schultz, A., Zöllner, R., Mogan, G. L. (2016).** A method for transforming electric vehicles to become autonomous vehicles. CONAT 2016 International Congress of Automotive and Transport Engineering, pp. 752–761. DOI: 10.1007/978-3-319-45447- 4_83.
8. **Han, X., Zhao, X., Xu, X., Mei, C., Xing, W., Wang, X. (2024).** Trajectory tracking control for underactuated autonomous vehicles via adaptive dynamic programming. Journal of the Franklin Institute, Vol. 361, No. 1, pp. 474–488. DOI: 10.1016/j.jfranklin.2023.12.003.
9. **Cruz, V. D., Rodriguez, J. A., Aguilar, L. T., Colorado, R. M. (2023).** Trajectory tracking control of wheeled mobile robots using neural networks and feedback control techniques. Studies in Computational Intelligence, pp. 381–393. DOI: 10.1007/978-3-031-28999-6_24.
10. **Hart, P., Rychly, L., Knoll, A. (2019).** Lane-merging using policy-based reinforcement learning and post-optimization. IEEE Intelligent Transportation Systems

Conference, pp. 3176–3181. DOI: 10.1109/itsc.2019.8917002.

11. **Bellusci, M., Cudrano, P., Mentasti, S., Cortelazzo, R. E. F., Matteucci, M. (2024).** Semantic interpretation of raw survey vehicle sensory data for lane-level hd map generation. Robotics and Autonomous Systems, Vol. 172, pp. 104513. DOI: 10.1016/j.robot.2023.104513.
12. **Han, Z., Gu, J., Feng, Y. (2023).** Blind lane detection and following for assistive navigation of vision impaired people. International Conference on Advanced Robotics and Mechatronics, pp. 721–726. DOI: 10.1109/icarm58088.2023.10218843.
13. **Chen, Y., Wong, P. K., Yang, Z. (2021).** A new adaptive region of interest extraction method for two-lane detection. International Journal of Automotive Technology, Vol. 22, No. 6, pp. 1631–1649. DOI: 10.1007/s12239-021-0141- 0.
14. **Haixia, L., Xizhou, L. (2021).** Flexible lane detection using CNNs. International Conference on Computer Technology and Media Convergence Design. DOI: 10.1109/ctmcd53128.2021.00057.
15. **Khan, M. A., Kee, S., Sikder, N., Mamun, M. A. A., Zohora, F. T., Hasan, M. T., Bairagi, A. K., Nahid, A. (2021).** A vision-based lane detection approach for autonomous vehicles using a convolutional neural network architecture. Joint 10th International Conference on Informatics, Electronics and Vision, pp. 1-10. DOI: 10.1109/ICIEVicIVPR52578.2021.9564229.
16. **Peregrina-Ochoa, S. A. (2019).** Sistema de navegación autónomo para un vehículo a escala mediante aprendizaje automático y visión por computadora. Tesis de Maestría en Ciencias en Ingeniería de Cómputo, Centro de Investigación en Computación, Instituto Politécnico Nacional.
17. **Neven, D., Brabandere, B. D., Georgoulis, S., Proesmans, M., Gool, L. V. (2018).** Towards end-to-end lane detection: an instance segmentation approach. IEEE Intelligent Vehicles Symposium, pp. 286–291. DOI: 10.1109/IVS.2018.8500547.
18. **Luca, A. D., Oriolo, G., Samson, C. (1998).** Feedback control of a nonholonomic car-like robot. Robot Motion Planning and Control, pp. 171–253. DOI: 10.1007/bfb0036073.
19. **Liu, D., Tang, M., Fu, J. (2022).** Robust adaptive trajectory tracking for wheeled mobile robots based on gaussian process regression. Systems and Control Letters, Vol. 163, pp. 105210. DOI: 10.1016/j.sysconle.2022.105210.
20. **Rosas-Vilchis, A. J. (2021).** Algoritmos de observación y control robustos para el vehículo autónomo. Tesis de Maestría en Ciencias en Sistemas Digitales, Centro de Investigación en Computación, Instituto Politécnico Nacional.
21. **Lu, Q., Chen, J., Wang, Q., Zhang, D., Sun, M., Su, C. (2022).** Practical fixed-time trajectory tracking control of constrained wheeled mobile robots with kinematic disturbances. ISA Transactions, Vol. 129, pp. 273–286. DOI: 10.1016/j.isatra.2021.12.039.
22. **Dixon, W. E., Dawson, D. M., Zergeroglu, E. (2000).** Tracking and regulation control of a mobile robot system with kinematic disturbances: a variable structure-like approach. Journal of Dynamic Systems, Measurement, and Control, Vol. 122, No. 4, pp. 616–623. DOI: 10.1115/1.1316795.
23. **Rodríguez-Arellano, J. A., Miranda-Colorado, R., Aguilar, L. T., Negrete-Villanueva, M. (2023).** Trajectory tracking nonlinear H∞ controller for wheeled mobile robots with disturbances observer. ISA Transactions, Vol. 142, pp. 372–385. DOI: 10.1016/j.isatra.2023.07.037.
24. **Cui, M., Liu, H., Wang, X., Liu, W. (2023).** Adaptive control for simultaneous tracking and stabilization of wheeled mobile robot with uncertainties. Journal of Intelligent and Robotic Systems, Vol. 108, No. 3 DOI: 10.1007/s10846-023-01908-0.
25. **Chang, S., Wang, Y., Zuo, Z., Yang, H., Luo, X. (2023).** Robust prescribed-time containment control for high-order uncertain multi-agent systems with extended state observer. Neurocomputing, Vol. 559, pp. 126782. DOI:10.16/j.neucom.2023.126782.

26. Github (2004). TuSimple Dataset. github.com/TuSimple/tusimple-benchmark.

27. Vu, D., Ngo, B., Phan, H. (2022). Hybridnets: end-to-end perception network. arXiv. DOI: 10.48550/ARXIV.2203.09035.

28. GitHub (2022). ONNX-hybridnets-multitaskroad- detection: Python scripts for performing road segmentation and car detection using the hybridnets multitask model in ONNX. github.com/ibaiGorordo/ONNX-HybridNets-Multitask-Road-Detection.

29. GitHub (2020). Pinto model repository for storing models that have been inter-converted between various models. github.com/PINTO 0309/PINTO_model_zoo

30. Marutotamtama, J. C., Setyawan, I. (2021). Physical distancing detection using YOLO v3 and bird's eye view transform. Proceedings of the 2nd International Conference on Innovative and Creative Information Technology, pp. 50–56. DOI: 10.1109/ICITech 50181.2021.9590157.

31. Sihombing, D. P., Nugroho, H. A., Wibirama, S. (2015). Perspective rectification in vehicle number plate recognition using 2d-2d transformation of planar homography. Proceedings of the International Conference on Science in Information Technology, pp. 237–240. DOI: 10.1109/ICSITech.2015. 7407810.

32. Yoo, S. J. (2013). Adaptive neural tracking and obstacle avoidance of uncertain mobile robots with unknown skidding and slipping. Information Sciences, Vol. 238, pp. 176–189. DOI: 10.1016/j.ins.2013.03.013.

33. Kairuz, R. I. V., Orlov, Y., Aguilar, L. T. (2021). Prescribed-time stabilization of controllable planar systems using switched state feedback. IEEE Control Systems Letters, Vol. 5, No. 6, pp. 2048–2053. DOI: 10.1109/LCSYS.2020.3046682.

34. Orlov, Y. (2020). Nonsmooth Lyapunov analysis in finite and infinite dimensions. Springer Cham. DOI: 10.1007/978-3-030-37625-3.

35. Biagiotti, L., Melchiorri, C. (2009). Trajectory planning for automatic machines and robots. Springer Berlin.

Article received on 22/01/2023; accepted on 24/04/2024.
**Corresponding author is Luis T. Aguilar.*

ISSN 2007-9737

Grammatical Evolution with Codons Selection Order as Intensification Process

Andrés Espinal, Marco Aurelio Sotelo-Figueroa*, Jorge Alberto Soria-Alcaraz

Universidad de Guanajuato,
Departamento de Estudios Organizacionales,
Mexico

masotelo@ugto.mx

Abstract. Grammar Evolution (GE) can be considered a form of Genetic Programming (GP) that has become very popular in the field of Automatic Programming (AP) over the last few years. There has been a lot of research on different aspects of GE, including its parts; the Search Engine, Mapping Process, and Grammar. However, it has been shown that it is possible to select the codons randomly to improve the GE, using a random permutation. This paper introduces a new approach to intensify a solution using permutation heuristics to guide the codon selection order. A non-parametric test was applied to discern between the results obtained by the proposal and those obtained by the canonical GE version and the GE with random permutations.

Keywords. Grammatical evolution, symbolic regression, intensification.

1 Introduction

Two important approaches to automatically generate computer programs are Genetic Programming (GP) [23, 26] and Automatic Programming (AP) [15, 3, 39]. These methods have proven invaluable in improving algorithms and resolving challenging issues.

They all have different restrictions and difficulties, and to solve some of these restrictions, a different evolution-inspired method appeared recently, namely Grammatical Evolution (GE) [37]. This method combines the strength of GP with a more methodical strategy based on formal grammar.

GP is an evolutionary algorithm that works based on a population of potential solutions, often represented as tree structures to develop programs [25, 31, 24]. By utilizing evolutionary operators including crossover, mutation, and selection, the goal of GP is to evolve programs that can carry out particular tasks.

Some of the issues GP can present in execution are program bloat, inefficiency, and a lack of control over the search space [32]. There are several proposals to improve GP [5, 10]; one of them [21] includes metaheuristics to improve GP performance.

By establishing a mapping mechanism that relates a given program genotype and phenotype, Grammatical Evolution (GE) offers a solution to these problems. It uses formal context-free grammars to do this [38]. Grammatical Evolution evolves strings of symbols as opposed to program trees directly.

A mapping process between these strings of symbols and computer programs is applied by this method, producing solutions with semantic and syntactic validity. The differentiation of genotype, the evolving string, from the phenotype, the completed program, is one of the major advances of Grammatical Evolution [34].

Because grammars may be created to contain specific structural and syntactical restrictions, this separation improves control over the search space by lowering the possibility that inappropriate or inefficient programs would be constructed.

Moreover, by altering the genotype, the search space may be explored more efficiently, producing

ISSN 2007-9737

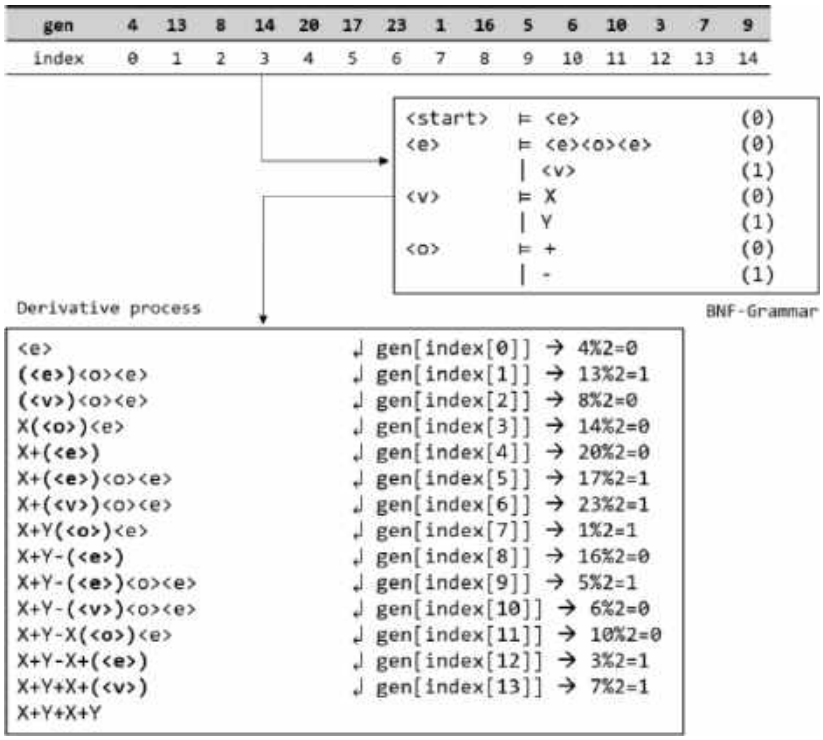


Fig. 1. Classical derivation example [45]

a more methodical program evolution. One of the research fields in GE is search engines.

There are several metaheuristics used as search engines like Genetic Algorithm (GA) [16], Differential Evolution (DE) [33, 17], Particle Swarm Optimization (PSO) [36, 42], Estimation Distribution Algorithms (EDA) [40, 29], and Ant Colony Optimization (ACO) [13] among others.

In [44], it was proposed to change the codon selection using a random permutation; this process was used with GE as a search engine, obtaining better results than classic GE. GE has been used successfully in many problems; however, the Symbolic Regression Problem (SRP) has been proposed and utilized as a benchmark problem for GE [1, 4, 43, 30].

The result of applying GE to SRP is an expression or function that fits a given instance of the problem. It is necessary to use formal grammar to ensure that the generated expression or function has syntactic validity.

Finally, GE explores a wide range of possible expressions at execution time, allowing it to discover highly accurate expressions. This paper proposes a methodology to evolve the codon selection order using the 2-opt, 3-opt, and 5-opt, and inversion permute heuristics.

This proposal is tested with instances of the SRP problem. The results obtained by this proposal, the canonical GE version, and the GE with Random Permutations were compared statistically.

2 Grammatical Evolution

The Grammatical Evolution (GE) [37, 38, 34] is a grammar-based form of Genetic Programming (GP) [22, 23]. The concepts of genotype and phenotype are present in both GP and GE.

Originally, the genotype in GP is based on tree representations, which are evaluated directly to obtain the phenotype, whereas GE uses a

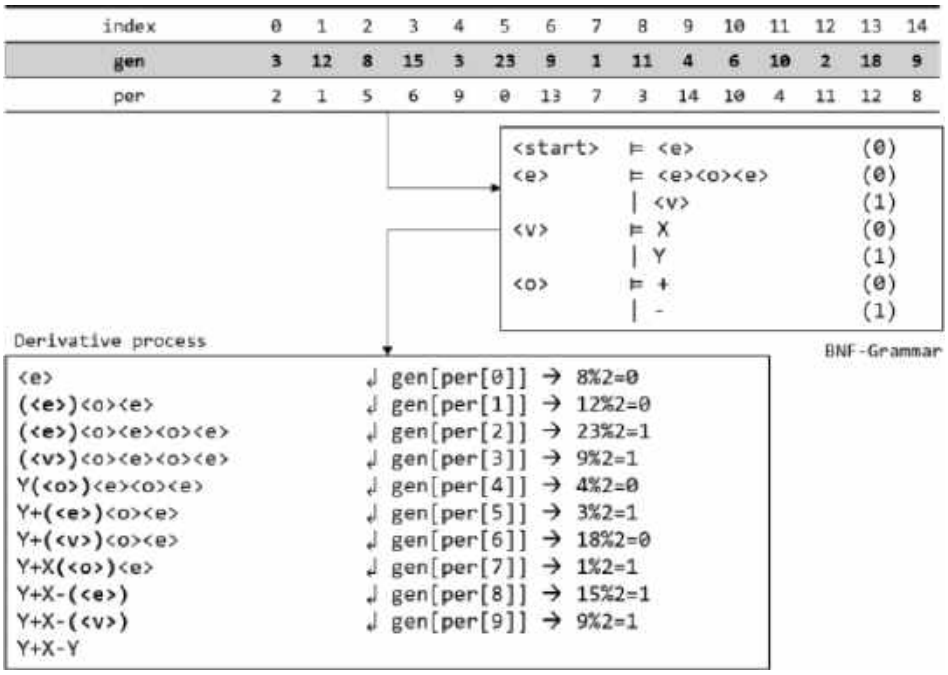


Fig. 2. Derivation example based on permutations [45]

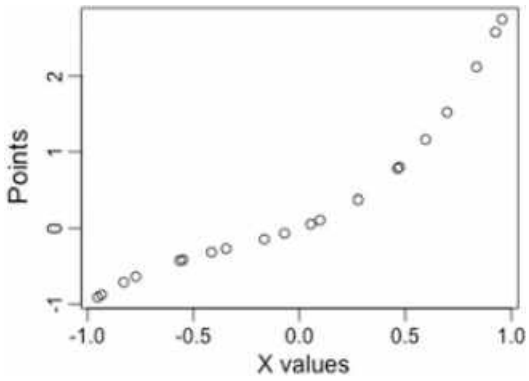


Fig. 3. Available data

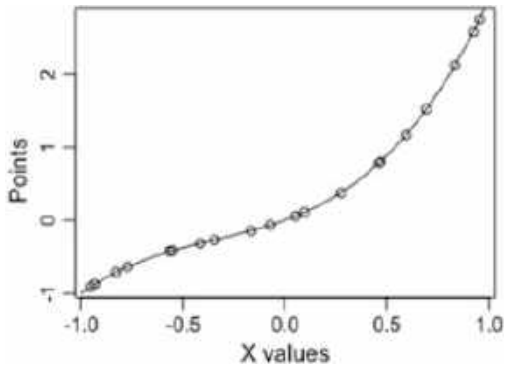


Fig. 4. Representative expression

linear representation, which is employed with a grammar to obtain the phenotype. The four main components, shown in Figure 5.

– The **Problem Instance** defines the problem domain and its conditions. It is used as a measure to guide the search engine and make the optimization process.

– The **Grammar** establishes elements and rules that fit the problem instance's specifics. In GE, Backus-Naur Form is commonly employed [2], although alternatives such as Attribute Grammar [20, 11] or Christiansen Grammar [7, 35] can also be applied.

– The **Search Engine** oversees optimization and adjusts the genotype based on the quality of the phenotype applied to the problem instance.

Algorithm 1 Grammatical evolution algorithm

Require: pop_size, dimtext, search_engine, cont_apply_LS, heuristic
1: Population ← new_population (pop_size, dim)
2: Per ← generate_permutation(dim)
3: Fitness ← Evaluate(Population, Per)
4: Cont ← 0
5: GBest ← get_Best(Population, Fitness)
6: **while** termination condition not met **do**
7: Population ← search_engine(Population)
8: Fitness ← Evaluate(Population, Per)
9: Best ← get_Best(Population, Fitness)
10: **if** GBest=Best **then**
11: Cont ← Cont + 1
12: **else**
13: GBest ← Best
14: Cont ← 0
15: **end if**
16: **if** Cont = cont_apply_LS **then**
17: Per ← heuristic(Per)
18: Cont ← 0
19: **end if**
20: **end while**
21: **return** Best Solution

While the Genetic Algorithm stands as the canonical search engine [16], numerous others have been implemented.

– The **Mapping Process** facilitates the conversion between genotype and phenotype, employing specific strategies such as Depth-First [34], Breadth-First [12], π Grammatical Evolution [36], etc.

Figure 1 shows an example of a classic GE, it uses a sequential codon selection order, and the search engine is applied to the codon values.

Figure 2 shows the proposal [44] example, it uses a permute codon selection order. Figure 1 and 2 uses the same codon values and the results show that is possible to obtain different phenotypes using a selection order.

3 Symbolic Regression Problem

Symbolic Regression Problem (SRP) [1, 4, 18] is the process of obtaining a representative expression from available data that we use when we want to know what was the equation behind our instance.

Table 1. Symbolic regression functions used as instances set

Function	Polynomial
F_1	$f(x) = X^3 + X^2 + X$
F_2	$f(x) = X^4 + X^3 + X^2 + X$
F_3	$f(x) = X^5 + X^4 + X^3 + X^2 + X$
F_4	$f(x) = X^6 + X^5 + X^4 + X^3 + X^2 + X$
F_5	$f(x) = sin(x^2)\, cos(x) - 1$
F_6	$f(x) = sin(x) + sin(x + x^2)$
F_7	$f(x) = log(x + 1) + log(x^2 + 1)$
F_8	$f(x) = \sqrt{x}$
F_9	$f(x, y) = sin(x) + sin(y^2)$
F_10	$f(x, y) = 2sin(x)\, cos(y)$
$Keijzer_1$	$f(x) = 0.3xsin(2\pi x)$
$Keijzer_2$	$f(x) = 1 + 3x + 3x^2 + x^3$
$Keijzer_3$	$f(x, y) = 8/(2 + x^2 + y^2)$
$Keijzer_4$	$f(x, y) = x^4 - x^3 + y^2/2 - y$
$Keijzer_5$	$f(x, y) = x^3/5 + y^3/2 - y - x$
$Keijzer_6$	$f(x_1, x_2, \ldots, x_10) = 10.59x_1x_2 + 100.5967x_3x_4 - 50.59x_5x_6 + 20x_1x_7x_9 + 5x_3x_6x_10$

In SRP, the goal is to seek a model (an equation or a mathematical formula) that describes the relationship between the input variables and the target output variable, without prior knowledge of the functional form.

SRP represents an important problem studied for the GP community [18, 43, 30].

Figure 3 shows the available data from the instance without knowing the function and Figure 4 shows the proposed expression to generate the data.

Table 2. Median of the results obtained for each instance and GE variant

Function	GA	Rnd	Inv	2opt	3opt	5opt
F_1	0.000	0.000	0.000	0.000	0.000	0.000
F_2	0.033	0.012	0.034	0.033	0.031	0.035
F_3	0.082	0.082	0.105	0.100	0.065	0.081
F_4	0.143	0.112	0.112	0.123	0.130	0.154
F_5	0.049	0.044	0.046	0.056	0.046	0.046
F_6	0.036	0.043	0.044	0.040	0.023	0.040
F_7	0.254	0.256	0.254	0.221	0.223	0.253
F_8	0.095	0.103	0.094	0.100	0.094	0.098
F_9	0.036	0.046	0.047	0.066	0.036	0.043
F_10	0.044	0.044	0.046	0.044	0.044	0.045
$Keijzer_1$	0.061	0.055	0.061	0.059	0.061	0.059
$Keijzer_2$	0.000	0.000	0.000	0.000	0.000	0.000
$Keijzer_3$	0.408	0.408	0.408	0.408	0.404	0.408
$Keijzer_4$	0.677	0.677	0.677	0.677	0.677	0.677
$Keijzer_5$	0.427	0.426	0.426	0.426	0.425	0.428
$Keijzer_6$	2.448	2.233	3.826	3.137	2.943	4.283

4 Proposed Approach

Figure 2 shows the proposal from [44] using a codon position random permutation. The proposal is shown in Algorithm 1, the permutation heuristics is applied if the best proposal solution is not improved for n generations.

In this study, the n value to apply the perturbation heuristic was determined empirically. The permutation heuristics were taken from the state-of-art based on those that can be applied to the Traveler Salesman Problem (TSP) [28, 8] and widely studied permutation problem.

4.1 k-opt

The k-opt heuristic [9, 27, 6] is classified as a local search method because it only slightly modifies the current solution to try to make it better.

It does not ensure the identification of the globally optimal solution, but it remains effective in perturbing an initial solution.

A k-opt heuristic is an attempt to enhance the quality of a solution using iterative swapping of pairs of items. Figure 6 shows an example of 2-opt.

Table 3. GA parameters

Parameter	Value
Population Size	300
Dimensions	100
Iterations to apply the heuristic	5
Function Evaluations	250,000
Selection	Binary Tournament
Crossover	2-points
Mutation	Bit-flip
Mapping Process	Depth-First

4.2 Inversion

The inversion heuristic [16, 8] involves selecting a subset of the proposal solution and investing the order of its elements. This heuristic neither ensures the identification of the globally optimal but is effective in perturbing an initial solution.

This operation introduces diversity in the population and can potentially explore different regions of the solution space. Figure 7 shows an example of Inversion.

5 Experimental Setup

Table 1 shows the Symbolic Regression functions used, the functions F_1 to F_{10} were taken from [18, 45] and the Keijzer from [19, 43, 14]. Mean Root Squared Error (MRSE), Equation 1, was used as a fitness function to discern the quality of each expression proposed by GE:

$$\mathrm{MRSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(t_i - y_i)^2}, \tag{1}$$

where:

– n is the number of data points.

– y_i is the real value.

– t_i corresponds to the value obtained.

Grammars used by each function [45] are the followings:

– Grammar 1 for functions F_1 to F_8.

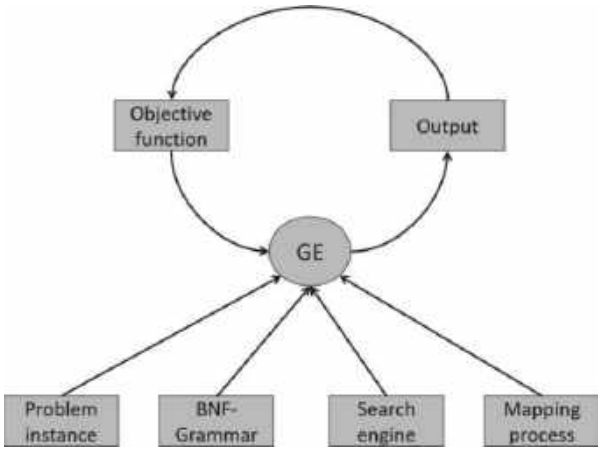


Fig. 5. GE methodology [41]

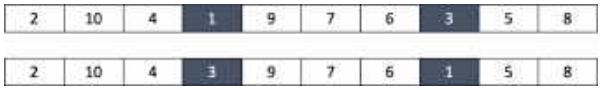


Fig. 6. 2-opt example, the first array contains a permutation from 1 to 10, it was chosen two items to interchange it into the second array

$$
\begin{aligned}
\langle \text{start} \rangle &::= \langle \text{expr} \rangle \\
\langle \text{expr} \rangle &::= (\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle) \mid \langle \text{pre} \rangle (\langle \text{expr} \rangle) \mid \langle \text{var} \rangle \\
\langle \text{var} \rangle &::= \texttt{1} \mid \texttt{x} \\
\langle \text{pre} \rangle &::= \texttt{exp} \mid \texttt{log} \mid \texttt{sin} \mid \texttt{cos} \\
\langle \text{op} \rangle &::= \texttt{*} \mid \texttt{/} \mid \texttt{+} \mid \texttt{-}
\end{aligned}
$$

Grammar 1. Grammar for the functions F_1 to F_8

– Grammar 2 for functions F_9 to F_{10}.

– Grammar 3 for functions $Keijzer_1$ to $Keijzer_5$.

– Grammar 4 for function $Keijzer_6$.

The parameters used in the classic GE, Random Permutation GE, and current proposal are shown in Table 3. Those parameters were taken from [44], and the new parameter used to apply the permutation heuristic was chosen empirically. To conduct the comparison, 33 individual runs were executed for each function, using the proposed approach, classical GE, and Random Permutation GE.

Table 4. Ranking based on medians

Algorithm	Ranking
3opt	2.28125
Rnd	3.03125
GA	3.53125
2opt	4.03125
5opt	4.03125
Inv	4.09375

$$
\begin{aligned}
\langle \text{start} \rangle &::= \langle \text{expr} \rangle \\
\langle \text{expr} \rangle &::= (\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle) \mid \langle \text{pre} \rangle (\langle \text{expr} \rangle) \mid \langle \text{var} \rangle \\
\langle \text{var} \rangle &::= \texttt{1} \mid \texttt{x} \mid \texttt{y} \\
\langle \text{pre} \rangle &::= \texttt{exp} \mid \texttt{log} \mid \texttt{sin} \mid \texttt{cos} \\
\langle \text{op} \rangle &::= \texttt{*} \mid \texttt{/} \mid \texttt{+} \mid \texttt{-}
\end{aligned}
$$

Grammar 2. Grammar for the functions F_9 to F_{10}

$$
\begin{aligned}
\langle \text{start} \rangle &::= \langle \text{expr} \rangle \\
\langle \text{expr} \rangle &::= (\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle) \mid \langle \text{var} \rangle \\
\langle \text{var} \rangle &::= \texttt{1} \mid \texttt{x} \mid \texttt{y} \\
\langle \text{op} \rangle &::= \texttt{*} \mid \texttt{/} \mid \texttt{+} \mid \texttt{-}
\end{aligned}
$$

Grammar 3. Grammar for the functions Keijzer from 1 to 5

$$
\begin{aligned}
\langle \text{start} \rangle &::= \langle \text{expr} \rangle \\
\langle \text{expr} \rangle &::= (\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle) \mid \langle \text{pre} \rangle (\langle \text{expr} \rangle) \mid \langle \text{var} \rangle \\
\langle \text{pre} \rangle &::= \texttt{exp} \mid \texttt{log} \mid \texttt{sin} \mid \texttt{cos} \\
\langle \text{var} \rangle &::= \texttt{1} \mid \texttt{x} \langle \text{c} \rangle \\
\langle \text{op} \rangle &::= \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/} \\
\langle \text{c} \rangle &::= \texttt{0} \mid \texttt{1} \mid \texttt{2} \mid \texttt{3} \mid \texttt{4} \mid \texttt{5} \mid \texttt{6} \mid \texttt{7} \mid \texttt{8} \mid \texttt{9}
\end{aligned}
$$

Grammar 4. Grammar for the function Keijze 6

The median of the results was used for statistical comparison among the proposed approach, GE, and Random Permutation GE.

The statistical test was performed using the non-parametric Friedman test, which aimed to establish if any implementation was capable of outperforming the others.

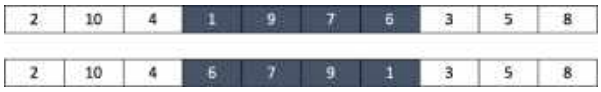


Fig. 7. Inversion example, the first array contains a permutation from 1 to 10, it was chosen a range to invert it into the second array

6 Results

Table 2 shows the median of the results of the 33 experiments for each instance. To discern between the results, a non-parametric Friedman test was performed. The value obtained was 11.99107143 with a p-value of 0.034910328.

With this p-value less than 0.1, it was possible to make a post-hoc procedure to determine which GE variant obtained the best results. The results of the post-hoc test are shown in Table 4.

7 Conclusions

This paper introduced a methodology to intensify the Grammatical Evolution solutions. The intensification was based on the codon position random permutation used previously, where it was shown that a random permutation improves the Grammatical Evolution results.

The permutation was guided by permutation heuristics from the state-of-the-art, using a k-opt and inversion heuristics that have been applied to permutation problems. Symbolic Regression Problems were used because they are widely used in Grammatical Evolution and Genetic Programming to analyze improvement and performance.

The results obtained using the proposal with a 3-opt heuristic are better than the random permutation and the classic Grammatical Evolution. Not all variations of the opt heuristic gave good results, nor did the inversion heuristic. The improvement of n generations was used as a parameter to apply the permutation heuristics; however, it is possible to identify a way to apply it without being an extra parameter.

References

1. **Augusto, D. A., Barbosa, H. J. (2000).** Symbolic regression via genetic programming. Proceedings of the VI Brazilian Symposium on Neural Networks, IEEE Computer Society, pp. 173–178. DOI: 10.1109/SBRN.2000.889734.
2. **Backus, J. W., Bauer, F. L., Green, J., Katz, C., McCarthy, J., Naur, P., Perlis, A. J., Rutishauser, H., Samelson, K., Vauquois, B., Wegstein, J. H., van-Wijngaarden, A., Woodger, M. (1963).** Revised report on the algorithmic language ALGOL 60. Communications of the ACM, Vol. 5, No. 4, pp. 349–367. DOI: 10.1093/comjnl/5.4.349.
3. **Balzer, R. (1985).** A 15 year perspective on automatic programming. IEEE Transactions on Software Engineering, Vol. SE-11, No. 11, pp. 1257–1268. DOI: 10.1109/TSE.1985.231877.
4. **Barmpalexis, P., Kachrimanis, K., Tsakonas, A., Georgarakis, E. (2011).** Symbolic regression via genetic programming in the optimization of a controlled release pharmaceutical formulation. Chemometrics and Intelligent Laboratory Systems, Vol. 107, No. 1, pp. 75–82. DOI: 10.1016/j.chemolab.2011.01.012.
5. **Brameier, M. F., Banzhaf, W. (2007).** Linear genetic programming. Springer US. DOI: 10.1007/978-0-387-31030-5.
6. **Brodowsky, U. A., Hougardy, S., Zhong, X. (2023).** The approximation ratio of the k-opt heuristic for the euclidean traveling salesman problem. SIAM Journal on Computing, Vol. 52, No. 4, pp. 841–864. DOI: 10.1137/21M146199X.
7. **Christiansen, H. (1990).** A survey of adaptable grammars. ACM SIGPLAN Notices, Vol. 25, No. 11, pp. 35–44. DOI: 10.1145/101356.101357.
8. **Cook, W. J., Applegate, D. L., Bixby, R. E., Chvátal, V. (2011).** The traveling salesman problem: A computational study. Princeton

University Press, Vol. 17. DOI: 10.1515/9781400841103.

9. **Croes, G. A. (1958).** A method for solving traveling-salesman problems. Operations Research, Vol. 6, No. 6, pp. 791–812. DOI: 10.1287/opre.6.6.791.
10. **Cárdenas-Florido, L., Trujillo, L., Hernandez, D. E., Muñoz-Contreras, J. M. (2024).** M5GP: Parallel multidimensional genetic programming with multidimensional populations for symbolic regression. Mathematical and Computational Applications, Vol. 29, No. 2, pp. 25. DOI: 10.3390/mca29020025.
11. **de-la-Cruz-Echeandía, M., de-la-Puente, A. O., Alfonseca, M. (2005).** Attribute grammar evolution. Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, Springer Berlin Heidelberg, pp. 182–191. DOI: 10.1007/11499305_19.
12. **Fagan, D., O'Neill, M., Galván-López, E., Brabazon, A., McGarraghy, S. (2014).** An analysis of genotype-phenotype maps in grammatical evolution. Esparcia-Alcázar, A.I., Ekárt, A., Silva, S., Dignum, S., Uyar, A.Ş. (eds) Genetic Programming. EuroGP 2010. EuroGP 2010. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, Vol. 60216021. DOI: 10.1007/978-3-642-12148-7_6.
13. **Gaddam, J., Barca, J. C., Nguyen, T. T., Angelova, M. (2023).** Grammatical evolution with adaptive building blocks for traffic light control. 2023 IEEE Congress on Evolutionary Computation (CEC), pp. 1–10. DOI: 10.1109/CEC53210.2023.10254190.
14. **Gupt, K. K., Raja, M. A., Murphy, A., Youssef, A., Ryan, C. (2022).** GELAB – The cutting edge of grammatical evolution. IEEE Access, pp. 1–1. DOI: 10.1109/ACCESS.2022.3166115.
15. **Hintze, G. (1966).** Automatic programming. Fundamentals of Digital Machine Computing, Springer Berlin Heidelberg, pp. 177–211. DOI: 10.1007/978-3-662-40151-4_7.
16. **Holland, J. H. (1992).** Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press.
17. **Indu, M. T., Shunmuga, V. C. (2024).** Differential evolution ensemble designer. Expert Systems with Applications: An International Journal, Vol. 238, No. C. DOI: 10.1016/j.eswa.2023.121674.
18. **Karaboga, D., Ozturk, C., Karaboga, N., Gorkemli, B. (2012).** Artificial bee colony programming for symbolic regression. Information Sciences, Vol. 209, pp. 1–15. DOI: 10.1016/j.ins.2012.05.002.
19. **Keijzer, M. (2003).** Improving symbolic regression with interval arithmetic and linear scaling. Genetic Programming, Springer, Berlin, Heidelberg, pp. 70–82. DOI: 10.1007/3-540-36599-0_7.
20. **Knuth, D. E. (1968).** Semantics of context-free languages. Mathematical systems theory, Vol. 2, No. 2, pp. 127–145. DOI: 10.1007/BF01692511.
21. **Korns, M. F. (2011).** Abstract expression grammar symbolic regression. Chapter 7, Springer New York, pp. 109–128. DOI: 10.1007/978-1-4419-7747-2_7.
22. **Koza, J. R. (1989).** Hierarchical genetic algorithms operating on populations of computer programs. IJCAI'89: Proceedings of the 11th international joint conference on Artificial intelligence, Vol. 1, pp. 768–774.
23. **Koza, J. R. (1992).** Genetic programming. Massachusetts Institute of Technology.
24. **Koza, J. R. (1994).** Genetic programming II: Automatic discovery of reusable programs, Vol. 1, The MIT Press. DOI: 10.5555/183460.
25. **Koza, J. R. (2010).** Human-competitive results produced by genetic programming. Genetic Programming and Evolvable Machines, Vol. 11, pp. 251–284. DOI: 10.1007/s10710-010-9112-3.

26. Langdon, W. B. (1998). Genetic programming and data structures: genetic programming+ data structures= automatic programming! Springer Science & Business Media. DOI: 10.1007/978-1-4615-5731-9.

27. Laporte, G., Gendreau, M., Potvin, J. Y., Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. International Transactions in Operational Research, Vol. 7, No. 4, pp. 285–300. DOI: 10.1016/S0969-6016(00)00003-4.

28. Larrañaga, P., Kuijpers, C., Murga, R. H., Inza, I., Dizdarevic, S. (1999). Genetic algorithms for the travelling salesman problem: A review of representations and operators. Artificial Intelligence Review, Vol. 13, No. 2, pp. 129–170. DOI: 10.1023/A:1006529012972.

29. Mégane, J., Lourenço, N., Machado, P., Schweim, D. (2023). The influence of probabilistic grammars on evolution. Proceedings of the Companion Conference on Genetic and Evolutionary Computation, Association for Computing Machinery, pp. 611–614. DOI: 10.1145/3583133.3590706.

30. Nicolau, M., Agapitos, A. (2021). Choosing function sets with better generalisation performance for symbolic regression models. Genetic Programming and Evolvable Machines, Vol. 22, No. 1, pp. 73–100. DOI: 10.1007/s10710-020-09391-4.

31. Nordin, P. (1999). Genetic programming III - Darwinian invention and problem solving. Evolutionary Computation, Vol. 7, No. 4, pp. 451–453. DOI: 10.1162/evco.1999.7.4.451.

32. O'Neil, M., Ryan, C. (2003). Grammatical evolution. Springer US. DOI: 10.1007/978-1-4615-0447-4_4.

33. O'Neill, M., Brabazon, A. (2006). Grammatical differential evolution. IC-AI, pp. 231–236.

34. O'Neill, M., Ryan, C. (2001). Grammatical evolution. IEEE Transactions on Evolutionary Computation, Vol. 5, No. 4, pp. 349–358. DOI: 10.1109/4235.942529.

35. Ortega, A., de-la-Cruz, M., Alfonseca, M. (2007). Christiansen grammar evolution: Grammatical evolution with semantics. IEEE Transactions on Evolutionary Computation, Vol. 11, No. 1, pp. 77–90. DOI: 10.1109/TEVC.2006.880327.

36. O'Neill, M., Brabazon, A. (2004). Grammatical swarm. Genetic and Evolutionary Computation–GECCO 2004: Genetic and Evolutionary Computation Conference, pp. 163–174.

37. Ryan, C., Collins, J. J., O'Neill, M. O. (1998). Grammatical evolution: Evolving programs for an arbitrary language. Genetic Programming: First European Workshop, pp. 89–96. DOI: 10.1007/BFb0055930.

38. Ryan, C., O'Neill, M., Collins, J. J. (2018). Handbook of grammatical evolution. Springer International Publishing. DOI: 10.1007/978-3-319-78717-6.

39. Schmid, U. (2003). Automatic programming. Inductive Synthesis of Functional Programs: Universal Planning, Folding of Finite Programs, and Schema Abstraction by Analogical Reasoning, pp. 99–166. DOI: 10.1007/978-3-540-44846-4_6.

40. Sotelo-Figueroa, M. A., Hernández-Aguirre, A., Espinal, A., Soria-Alcaraz, J. A., Ortiz-López, J. (2018). Symbolic regression by means of grammatical evolution with estimation distribution algorithms as search engine. Fuzzy Logic Augmentation of Neural and Optimization Algorithms: Theoretical Aspects and Real Applications, Springer International Publishing, pp. 169–177. DOI: 10.1007/978-3-319-71008-2_14.

41. Sotelo-Figueroa, M. A., Puga-Soberanes, H. J., Carpio-Valadez, J. M., Fraire-Huacuja, H. J., Cruz-Reyes, L., Soria-Alcaraz, J. A. (2014). Improving the bin packing heuristic through grammatical evolution based on swarm intelligence. Mathematical Problems in

Engineering, Vol. 2014, No. 1, pp. 1–12. DOI: 10.1155/2014/545191.

42. **Tsoulos, I. G., Tzallas, A. (2023).** A feature construction method that combines particle swarm optimization and grammatical evolution. Applied Sciences, Vol. 13, No. 14, pp. 8124. DOI: 10.3390/app13148124.

43. **White, D. R., McDermott, J., Castelli, M., Manzoni, L., Goldman, B. W., Kronberger, G., Jaśkowski, W., O'Reilly, U. M., Luke, S. (2013).** Better GP benchmarks: Community survey results and proposals. Genetic Programming and Evolvable Machines, Vol. 14, No. 1, pp. 3–29. DOI: 10.1007/s10710-012-9177-2.

44. **Zúñiga, B. V., Carpio, J. M., Sotelo-Figueroa, M. A., Espinal, A., Purata-Sifuentes, O. J., Ornelas, M., Soria-Alcaraz, J. A., Rojas, A. (2020).** Exploring random permutations effects on the mapping process for grammatical evolution. Journal of Automation, Mobile Robotics and Intelligent Systems, pp. 65–72. DOI: 10.14313/JAMRIS/1-2020/8.

45. **Zuñiga-Nuñez, B. V., Carpio, J. M., Sotelo-Figueroa, M. A., Soria-Alcaraz, J. A., Purata-Sifuentes, O. J., Ornelas, M., Rojas-Domínguez, A. (2020).** Studying grammatical evolution's mapping processes for symbolic regression problems. Intuitionistic and Type-2 Fuzzy Logic Enhancements in Neural and Optimization Algorithms: Theory and Applications, Springer, pp. 445–459. DOI: 10.1007/978-3-030-35445-9_32.

Article received on 30/01/2024; accepted on 06/05/2024.
**Corresponding author is Marco Aurelio Sotelo-Figueroa.*

ISSN 2007-9737

Bird Swarm Algorithm and Particle Swarm Optimization in Ensemble Recurrent Neural Networks Optimization for Time Series Prediction

Martha Pulido, Patricia Melin*

Tecnológico Nacional de México, Campus Tijuana,
Mexico

{martha.pulido, pmelin}@tectijuana.mx

Abstract. Recurrent Neural Networks have proven to provide good results in time series prediction. In this paper, an ensemble recurrent neural network, for prediction is used. Euro/Mexican pesos and Euro/Dollar series are utilized to develop a prediction model. The design of this consists of an ensemble recurrent neural network and the optimization of its structure is achieved with the bird swarm algorithm and Particle Swarm Optimization. The outputs of the networks are integrated with type-1 and type-2 fuzzy systems. These fuzzy systems are of the Mamdani type. The tests were realized with the designed method and a good result was obtained in the application to time series, as well as a comparison of the two optimization algorithms was made.

Keywords. Bird swarm algorithm, time series, recurrent neural networks, optimization, prediction.

1 Introduction

In nature, we find situations and processes for which there is no immediate explanation. A particular case of this is swarming, in which many simple individuals act in such a way that intelligent behavior emerges.

Swarm intelligence is the branch of artificial intelligence that seeks to study and simulate the complex behaviors found in these swarms, such as flocks of birds or ant colonies. In this case, an algorithm based on the behavior of birds was implemented, applying it to complex time series prediction.

A time series can represent anything from the prices of an item, unemployment rates, the maximum daily temperature, and wind speed, to the efforts and temperatures at various points of an instrumented civil work, etc., since it is a sequence of data arranged chronologically and are used to study the causal relationship between various variables that change over time.

In this work, we use recurrent neural networks (RNNs) since these are one of the most used models. For example, in voice recognition systems or video analysis, or in natural language processing, and these models are these models capable of processing different types of sequences (such as videos, conversations, text) and, unlike neural or convolutional networks, this type of neural network are capable of generating new sequences, likes time series.

RNNs use the concept of recurrence: to generate the output, which we will call activation from now on. The network uses not only the current input, but also the activation generated in the previous iteration. In a nutshell, RNNs use a certain type of “memory” to generate the desired output [17].

Some models based on RNNs can be found in finance [4], electronic commerce [5, 6], capital markets [7, 8], macroeconomy [9, 10], health [11, 12], signal processing, meteorology [13, 14], voice recognition [15] and traffic control [16].

When birds are fed with flocks of birds, we can have more information and a good foraging effectiveness and survival benefits.

Table 1. Parameters of the BSA

Parameter	Value
M1	100
pop	100
Dime	60
FQ1	9
Co1	0.5
Co2	0.5
a1	2
a2	2

Table 2. Parameters of the PSO

Parameters	Value
P	100
I	100
C1	2
C2	2
C	0-0.9
W	0-0.9

Table 3. Parameters

Parameters ERRN	Value	
	Min	Max
NN1	1	5
NL1	1	3
NN1	1	30

Table 4. Euro/Mexican pesos time series results of the optimization

No.	NM	NL	Time	PE
1	2	16, 16 18, 17	01:38:47	0.01806
2	2	10 15	01:40:41	0.01842
3	2	24, 22 16, 23	01:35:11	0.019807
4	2	25, 16 1, 6	01:40:41	0.01802
5	2	25, 17 15, 16	01:15:42	0.01827
6	2	6, 9 28, 4	01:45:35	0.019806
7	2	6, 9 28, 4	01:35:35	0.01827
8	1	17	01:42:17	0.01822
9	2	25 15	01:15:42	0.01835
10	2	6.9, 22 26, 2, 26	01:26:15	0.01807

If one bird finds food, others may feed from it [19] While foraging, birds often unite due to the Predation threat [20]. They constantly scan their surroundings and raise their heads.

These behaviors, interpreted as vigilance behavior [21], may be conducive to detecting Predators [22]. Studies showed that birds would randomly choose between foraging and keeping vigilance [23]. Birds often give alarm calls when they detect a Predator [24]. Thus, the whole group would fly off together. Birds in the border of a group have higher probability of being attacked by Predators than those in the center.

Studies suggest that animals foraging in the flock center may move to their neighbors to protect themselves from being attacked [25]. The birds move towards the center of the flock as they perceive them. This motion, however, may be affected by the interference induced by competing bird swarms [26].

For this reason, the birds move toward the center of the swarm. This work outlines the formation of the ensemble recurrent neural network (ERNN). This model is applied to time series prediction [27, 32].

The architecture is optimized with the bird swarm algorithm, (BSA) [29-32], and Particle Swarm Optimization (PSO). The responses of the ERNN modules are integrated with type-1 and type-2 fuzzy systems (IT2FS), [32-38]. The optimization of the RNN consists finding the best the number of hidden layer (NL), number of neurons (NN) and the number of modules (NM).

We integrate responses in the ERNN, with type-1 and Fuzzy System Type-2 (IT2FS) and in this way we achieve Prediction. The Mamdani fuzzy inference system (FIS) has five inputs (Pr1, Pr2, Pr3, Pr4, and Pr5) and one output that is called Prediction.

The number of inputs of the fuzzy system (FS) is according to the outputs of ERNN and Mamdani fuzzy FIS is created. This FIS five inputs, which are Pr1, Pr2, Pr3, Pr4, and Pr5, have a from the range from 0 to 1.4.

The output is called Prediction, the range goes from 0 to 1.4 and is granulated into two membership functions (MFs) “Low”, “High” [50-53]. The main contribution of this work is the

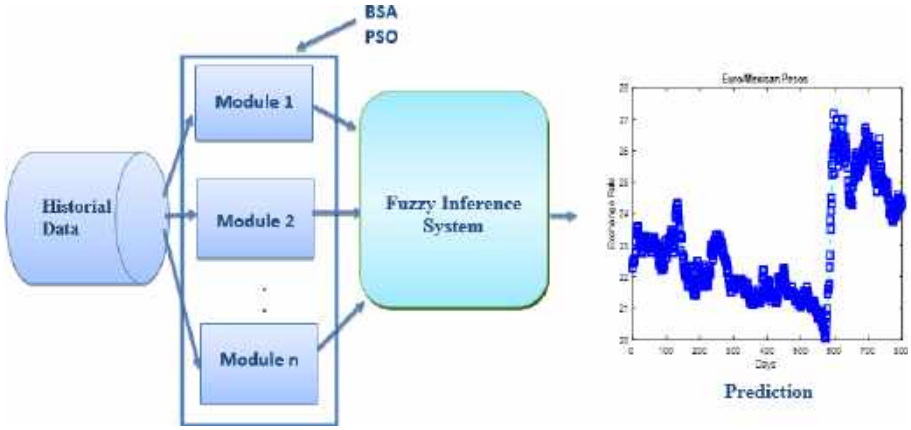


Fig. 1. General scheme

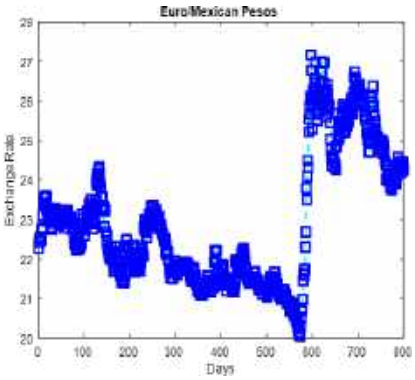


Fig. 2. Euro/Mexican pesos time series

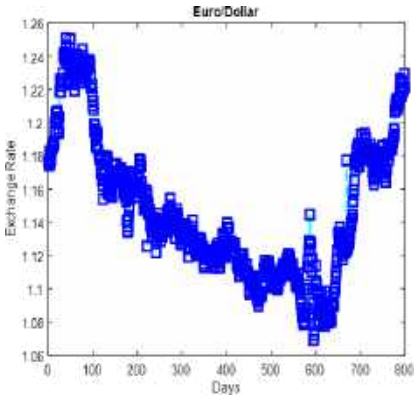


Fig. 3. Euro/Dollar time series

optimization of recurrent neural networks with the BSA algorithm, and PSO applied to time series, which represents a different way to optimize this kind of neural networks.

This paper is consisting of 4 parts: In section 2, the problem to be solved and the solution with the proposed method are mentioned, section 3 describes in the results obtained with the model and section 5 offers the conclusions.

2 Proposed Method

In this section, a recurrent neural network optimization model with the Bird Swarm algorithm (BSA) is offered. We have historical data that enters the ENN modules, and then responses from the ENN are integrated with type-1 and IT2FS for the Prediction of the time series.

In this case: Euro/Mexican pesos and Euro/Dollar time series. Figure 1 shows a general scheme of each of the steps of the method postulated in this paper which is explained in more detail below:

2.1 Parameters for BSA Applied to the RNN

Table 1 summarizes the parameters used for the optimization of ERNN with the BSA, where Dime is the number of dimensions used for optimization, M1 is the iteration number, pop is the bird population, FQ1 (FL) is the bird behavior frequency, a1 and a2 are values related to the vigilance behavior of the birds these effects are

If (Pr1 is Pe1L) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4L) and (Pr5 is Pe5H) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5L) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5H) then (Pr is Lo) If (P1 is Pe1L) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4L) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is P5eH) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4L) and (Pr5 is P5eH) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5L) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5H) then (Pr is Hh) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3H) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Lo) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is Pe5H) then (Pr is Lo) If (Pr1 is Pe1L) and (Pr2 is Pe2H) and (Pr3 is Pe3L) and (Pr4 is Pe4H) and (Pr5 is P5eH) then (Pr is Hh) If (Pr1 is Pe1H) and (Pr2 is Pe2L) and (Pr3 is Pe3H) and (Pr4 is Pe4L) and (Pr5 is Pe5L) then (Pr is Lo)

Fig. 4. IT2FS rules

Table 5. Results integration for the Euro/Mexican pesos

No.	Type-1 Fuzzy Integrator
1	0.3026
2	0.1408
3	0.1384
4	0.1122
5	0.3175
6	0.3022
7	0.1408
8	0.1383
9	0.1383
10	0.3021

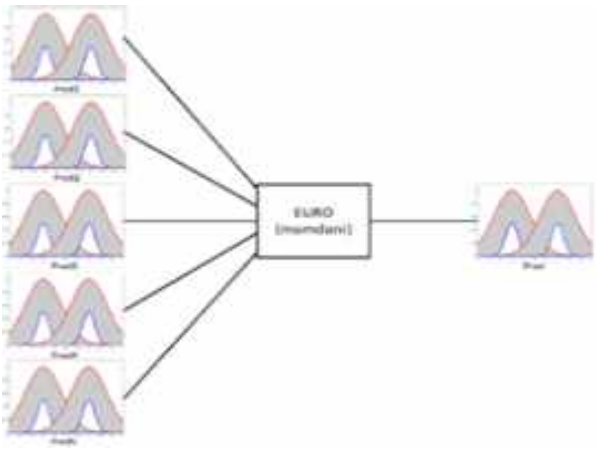


Fig. 5. IT2FS

indirect and direct. co1 cognitive is the coefficient and co2 is the accelerated social coefficient. Table 2 illustrated the parameters used for the optimization of ERNN with the PSO, where P is the number of particles, I is the iteration number, the cognitive component is C1, the Social Component is C2, Constriction coefficient of linear increase is Cand Inertia weight with linear decrease is (W).

2.2 Parameters for PSO Applied to and RNN

The following equation is used to minimize the prediction error, the objective function we

Table 6. Results for the IT2FS for the Euro/Mexican pesos

No.	PE 0.3	PE 0.4	PE 0.5
1	0.2219	0.2210	0.2286
2	0.2263	0.2268	0.2274
3	0.2215	0.2236	0.2339
4	0.2212	0.2222	0.2228
5	0.2229	0.2339	0.2348
6	0.2222	0.2225	0.2231
7	0.2221	0.2229	0.2235
8	0.2261	0.2268	0.2371
9	0.2256	0.2268	0.2387
10	0.2268	0.2274	0.2389

Table 7. Euro/Dollar pesos time series results of the optimization

No.	NM	NL	Time	PE
1	**3**	18 15 17	01:05:22	0.00042061
2	**3**	5, 18 15, 13 17, 18	01:19:33	0.00055028
3	**4**	17, 5, 7 12, 25, 13 20, 27, 22	01:28:04	0.00042102
4	**2**	10, 11 11, 12 15, 17 18, 16	01:45:20	0.0005802
5	**3**	18, 19, 17 13, 13.18	01:10:15	0.00050568
6	**3**	22, 19, 14 27, 17, 12 16, 17, 19	01:11:03	0.00051168
7	**3**	8, 22, 21 11, 23, 21 16, 6, 23	01:33:18	0.00052428
8	**3**	18, 19, 25 15, 22, 19 16, 11, 23	01:22:20	0.00528091
9	**4**	7, 7, 12 21, 2, 19 21, 13, 12	01:51:21	0.00505413
10	**4**	5, 27, 28 20, 20, 6 13, 12, 12 12, 26, 6	01:51:22	0.00042107

employed with the bird swarm algorithm and PSO:

$$\mathrm{ERMod} = \frac{1}{d}\sum_{i=1}^{d}|pd_i - rd_i|, \tag{1}$$

$$\mathrm{PE} = \frac{1}{T}(\mathrm{ERMod}_1 + \mathrm{ERMod}_2 + \ldots + \mathrm{ERMod}_N), \tag{2}$$

where PE is Prediction Error, d the data number, pd corresponds to the predicted data for each of the 5 modules, rd represents the real data, ERMod is the Module Prediction error, T represent module number determined by the BSA and the Predicion Error is the average prediction error for the modules. Table 2 represents the search space parameters.

2.3 Data Base

Fig. 2 presents the plot of the data Euro/Mexican pesos [18], where we use 800 data that belong to the time of 12/08/2017 to 31/12/2020. We apply 30% to test the RNN and 70% for the training of the RNN. Fig. 3 exhibits the plot of the data Euro/Dollar [18], where we use 800 data that belong to the time 12/08/2017 to 31/12/2020. We apply 30% to test the RNN and 70% for the training of the RNN.

2.4 Description of the Architecture of Type-1 and IT2FS

The following equation shows how the Fuzzy System is calculated:

$$y = \frac{\sum_{i=1d_i}^{n} m\left(d_i\right)}{\sum_{i=1}^{n} m\left(d_i\right)_i}, \tag{3}$$

where m corresponds MFs and represents the input data.

Table 8. Results of the integration for Euro/Dollar

No.	Type-1 fuzzy integrator
1	0.1225
2	0.1228
3	0.1432
4	0.2223
5	0.225
6	0.2822
7	0.1736
8	0.3220
9	0.2217
10	0.2325

Table 9. Euro/Mexican pesos time series results of the PSO

No	Modules	Layers	Time	P
1	3	16,18 20,22 15,17	01:07:17	0.01922
2	2	22,23 24.25	01:02:23	0.02202
3	3	24,24 23,21 19.20	01:05:11	0.01988
4	2	27,21 24,23	01:07:22	0.02107
5	4	27,15 20,21 25,24 23,22	01:07:25	0.01977
6	2	26,29 28,27	01:07:32	0.02106
7	3	6,9 28,4	01:35:35	0.01827
8	2	20,23 21,22	01:07:17	0.01738
9	2	25,23 21,22	01:07:12	0.01732
10	3	6,9,22 26,2,26 22,21,10	01:06:13	

2.5 Type-2 Memberships Functions

Below are equations of type-2 fuzzy system membership functions:

$$\widetilde{u}(x) = [\mu(x), \widetilde{u}(x)] = \text{gausstype2}(x, [\sigma_x m_1, m_2]), \quad (4)$$

where "igaussstype2" stands for the Gaussian generalized type-2 membership function with uncertain mean:

$$m_x = \frac{m_1 + m_2}{2}, \quad (5)$$

$$p_x = \text{gaussmf}(x, [\sigma_x m_1, m_2]), \quad (6)$$

$$\implies p_x = \exp\left[\frac{1}{2}\left(\frac{x - m_x}{\sigma_x}\right)^2\right], \quad (7)$$

$$p_x = \text{gaussmf}(x, [\sigma_x, m_x]), \quad (8)$$

$$\widetilde{u}(x, \mu) = \text{gaussmf}(\mu, [\sigma_x, p_x]), \quad (9)$$

$$\implies \widetilde{u}(x, \mu) = \exp\left[\frac{1}{2}\left(\frac{x - p_x}{\sigma_u}\right)^2\right], \quad (10)$$

where $p_x =$ is of the center of the function and σ_u is the implication of the function. MF too high:

$$\text{MFToohigh} = e^{-\frac{1}{2}\left(\frac{x - 234}{25.2}\right)^2}, \quad (11)$$

where $p_x = 234$ and $\sigma_u = 25.2$, where $p_x = 234$ and $\sigma_u = 25.2$. The fuzzy system that was used has five inputs that are Pr1, Pr2, Pr3, Pr4, and Pr5, and an output called Pr, where the parameters have a range from 0 to 1.4 and the variables are denominated "Low" and "High", in Figure 5 as present. The Figure 4 lists the rules used in the type-1 and IT2FL.

The rules were obtained based on previous work used in the fuzzy system [36]. The Figure 4 represents the rules employed in the fuzzy system. The rules were formulated based on previous work used in the fuzzy system where 32 rules were used and they are all possible rules since we have 5 inputs and one output, and each input has two MFs and the consequents are determined by trial and error [36].

Table 10. Results for the IT2FS for the Euro/Dollar

No.	PE 0.3	PE 0.4	PE 0.5
1	0.3827	0.3849	0.3864
2	0.4162	0.4165	0.4168
3	0.4168	0.4236	0.4238
4	0.4229	0.4239	0.4249
5	0.4422	0.4436	0.4443
6	0.4987	0.4222	0.4229
7	0.4668	0.4544	0.4277
8	0.4492	0.4898	0.4899
9	0.4449	0.4453	0.4456
10	0.4966	0.4988	0.4994

Table 11. Results integration for the Euro/Mexican pesos for PSO

No.	Type-1 fuzzy integrator
1	0.3228
2	0.3417
3	0.3263
4	0.3123
5	0.3442
6	0.3227
7	0.3409
8	0.3386
9	0.3323
10	0.3122

Table 12. Results for the IT2FS for the Euro/Mexicans pesos for PSO

No.	PE 0.3	PE 0.4	PE 0.5
1	0.2330	0.2332	0.2377
2	0.2383	0.2385	0.2366
3	0.2318	0.2338	0.2380
4	0.2333	0.2346	0.2348
5	0.2388	0.2389	0.2391
6	0.2382	0.2385	0.2381
7	0.2371	0.2373	0.2376
8	0.2381	0.2384	0.2387
9	0.2390	0.2392	0.2396
10	0.2377	0.2379	0.2381

3 Results

In this section, the results of the experiments of the presented model are outlined. In this case, the bird algorithm and particle swarm optimization were used for the optimization of the ERNN, and the type-1 and IT2FS integration and 10 experiments were carried out and the time series of the Euro/Mexican pesos and Euro/Dollar pesos were considered. Table 4 represents the Euro/Mexican pesos time series results of the Optimization of ERNN for 10 experiments, where 100 generations and 100 population is used.

Table 5 shows the Euro/Mexican pesos time series results of 10 type-1 fuzzy system experiments. Table 6 represents the Euro/Mexican pesos time series results of the IT2FLS for 10 experiments with different levels of uncertainty.

Table 7 represents Euro/Dollar time series results of the Optimization of ERNN for 10 experiments, where 100 generations and 100 population is used. Table 8 shows the Euro/Dollar time series results of 10 type-1 fuzzy system experiments. Table 9 represents the Euro/Mexican pesos time series results of the PSO of ERNN 10 experiments, where 100 particles and 100 iterations is used. Table 10 represents the Euro/Dollar time series results of 10 IT2FS experiments with different uncertainty levels.

Table 11 shows the Euro/Mexican pesos time series results for 10 type-1 fuzzy system experiments for PSO. Table 12 represents the Euro/Mexican pesos time series results of the IT2FLS 10 experiment for PSO with different levels of uncertainty.

Table 13 shows the Euro/Dollar time series results of 10 type-1 fuzzy system experiments for PSO. Table 14 represents the Euro/Dollar time series results of 10 IT2FS experiments for PSO with different levels of uncertainty.

To compare the two optimization algorithms, Bird Sarm Algorithm and Particle Swarm, for Euro/Mexican pesos time series, the t statistical test was used and we can conclude that there is significant evidence between the optimization of the NN with the BSA and PSO. In other words, it can be said that the BSA algorithm is better.

Table 13. Euro/Dollar pesos time series results of the PSO

No.	Mod.	Layers	Time	P
1	2	15,17 15,16	01:08:35	0.00052058
2	2	17,15 15,15	01:08:33	0.00051022
3	3	16,15,15 13,15,16 15,17,12	01:08:31	0.00052323
4	3	21,20,19 22,18,20 21,20,18 20,22,22	01:08:27	0.00062023
5	2	15,18,19 16,15,17	01:08:26	0.00056032
6	4	21,20,19 22,18,20 21,20,18 20,22,22	01:09:03	0.00051168
7	3	21,22, 15,16 16,17	01:08:55	0.00053328
8	4	20,19 21,22	01:08:28	0.00628082
9	2	21,20 22,18 21,20 20,22	01:09:01	0.00605332
10	3	2,27 20,21 17,20	01:09:09	0.00661133

Table 14. Results of the integration for Euro/Dollar for PSO

No.	PE 0.3	PE 0.4	PE 0.5
1	0.4327	0.4449	0.4451
2	0.4522	0.4523	0.4528
3	0.4238	0.4242	0.4244
4	0.4266	0.4268	0.4270
5	0.4292	0.4298	0.4301
6	0.4287	0.4289	0.4290
7	0.4266	0.4268	0.4270
8	0.4292	0.4294	0.4296
9	0.4275	0.4277	0.4279
10	0.4223	0.4226	0.4229

Table 15. t-Statistical test for Euro/Mexican pesos

Algorithm	N	Mean	Standard Deviation	Mean Error
BSA	29	0.018518	0.00635	0.00012
PSO	29	0.01955	0.00154	0.00029

Table 16. t-Statistical test for Euro/Dollar

Algorithm	N	Mean	Standard deviation	Mean Error
BSA	29	0.00355	0.00927	0.0017
PSO	29	0.00261	0.00261	0.00049

The number of samples was 29, and a 95% confidence interval was used. and produce a P value of 0.002 and T value of -3.34 and Critical Value of 0.0668. To compare the two optimization algorithms of BSA and PSO for Euro/Dollar Peso time series, t-statistic test was used and we can conclude that there is no significant evidence between the optimization of with the BSA and PSO algorithm. In other words, it can be said that the BSA algorithm and the PSO are equal since the results are almost the same, the samples used were 29 and a 95% confidence interval was used.

4 Conclusions

Recurrent neural networks are an excellent model for time series data. In this paper we, so we conclude that this type of network produces good results, as well as the optimization of the structure of this network, and then the fuzzy integration of the network responses to test this method was used Euro/Dollar and Euro/Mexican pesos time series. When performing statistical tests, we had the goal finding out it both algorithms are good at solving time series problems.

In the Euro/Mexican time series the BSA algorithm was better and for the Euro/Dollar time series, both were goodComparisons were made with the two optimization algorithms Bird Swarm Algorithm and Particle Swarm, for Euro/Mexican pesos time series, the statistical t test was used and we can conclude that there is significant evidence between the optimization with the BSA and PSO.

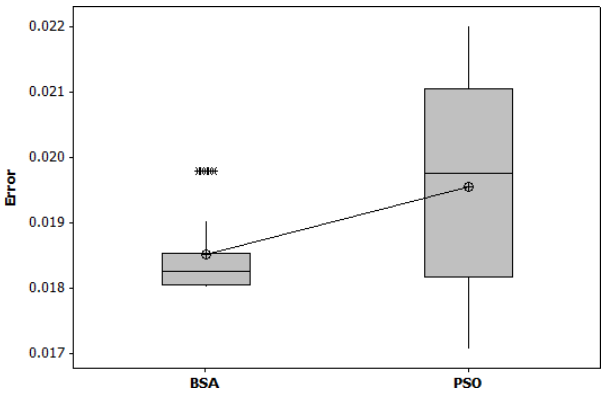


Fig. 6. Box plot for Euro/Mexican pesos time series

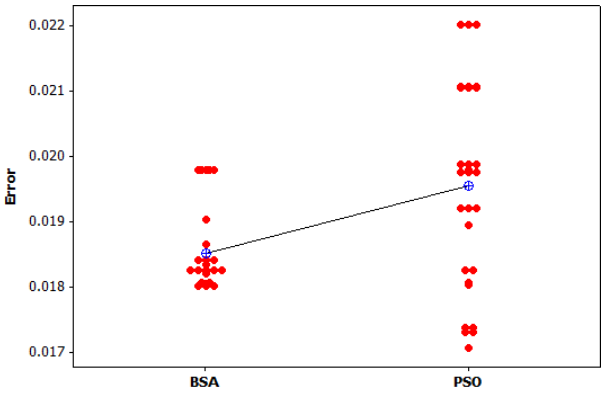


Fig. 7. Graph of values for Euro/Mexican pesos time series

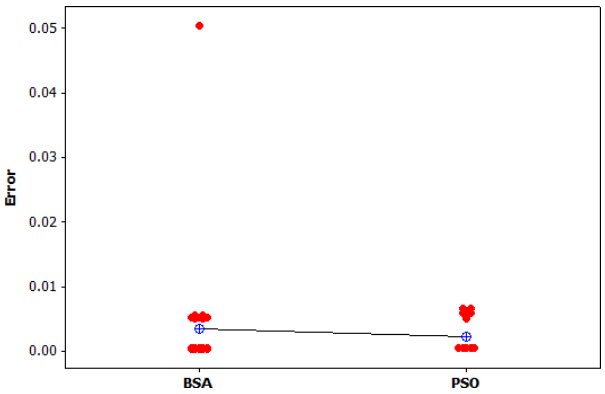


Fig. 8. Graph of values for Euro/Dollar time series

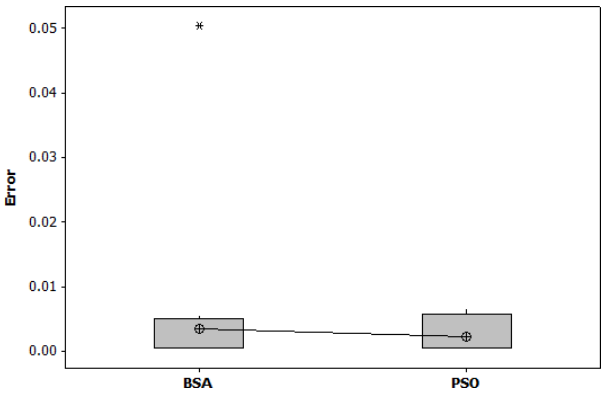


Fig. 9. Graph of values for Euro/Mexican pesos time series

In other words, it can be said that the BSA algorithm is better, and for Euro/Dollar Peso time series, the t-statistic test was also used and we can conclude that there is no significant evidence between the optimization with the BSA algorithm and PSO. In other words, it can be said that the BSA algorithm and the PSO are the same. As future work, we plan to consider other similar metaheuristics (instead of BSA and PSO), like in [53-57], and also undertake other application areas [58-61].

Acknowledgments

We would like to express our gratitude to CONAHCYT and Tijuana Institute of Technology for the resources granted for this research.

References

1. **Anderson, T. R. (2002).** Biology of the ubiquitous house parrow: From genes to Populations. Oxford University Press.

2. **Ahmad, M., Javaid, N., Niaz, I. A., Shafiq, S., Rehman, O. U., Hussain, H. M. (2018).** Application of bird swarm algorithm for solution of optimal power flow problems. Complex, Intelligent, and Software Intensive Systems, pp. 280–291. DOI: 10.1007/978-3-319-93659-8_25.

3. **Gad, A. G. (2022).** Particle swarm optimization algorithm and its applications: A systematic review. Archives of Computational Methods in Engineering, Vol. 29, No. 5, pp. 2531–2561. DOI: 10.1007/s11831-021-09694-4.

4. **Beauchamp, G. (2003).** Group-size effects on vigilance: A search for mechanisms. Behavioural Processes, Vol. 63, No. 3, pp. 111–121. DOI: 10.1016/s0376-6357(03)00002-0.

5. **Bednekoff, P. A., Lima, S. L. (1998).** Randomness, chaos and confusion in the study of antipredator vigilance. Trends in Ecology and Evolution, Vol. 13, No. 7, pp. 284–287. DOI: 10.1016/s0169-5347(98)01327-5.

6. **Brockwell, P. J., Davis R. A. (2002).** Introduction to time series and forecasting, Springer Texts in Statistics. DOI: 10.1007/0-387-21657-X_8.

7. **Cai, L., Zhang, Y., Ji, W. (2018).** Variable strength combinatorial test data generation using enhanced bird swarm algorithm. Proceedings of the 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, pp. 391–398. DOI: 10.1109/snpd.2018.8441104.

8. **Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y. (2018).** Recurrent neural networks for multivariate time series with missing values. Scientific Reports, Vol. 8, No. 1, pp. 6085. DOI: 10.1038/s41598-018-24271-9.

9. **Correa, P. P., Cipriano, A., Nunez, F., Salas, J. C., Lobel, H. (2021).** Forecasting copper electrorefining cathode rejection by means of recurrent neural networks with attention mechanism. IEEE Access, Vol. 9, pp. 79080–79088. DOI: 10.1109/access.2021.3074780.

10. **Castillo, O., Melin, P. (2007).** Comparison of hybrid intelligent systems, neural networks and interval type-2 fuzzy logic for time series prediction. Proceedings of the International Joint Conference on Neural Networks, pp. 3086–3091. DOI: 10.1109/ijcnn.2007.4371453.

11. **Castillo, O., Melin, P. (2002).** Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic, and fractal theory. IEEE Transactions on Neural Networks, Vol. 13, No. 6, pp. 1395–1408. DOI: 10.1109/tnn.2002.804316.

12. **Castro, J. R., Castillo, O., Melin, P., Mendoza, O., Rodríguez-Díaz, A. (2010).** An interval type-2 fuzzy neural network for chaotic time series prediction with cross-validation and akaike test. Soft Computing for Intelligent Control and Mobile Robotics, pp. 269–285. DOI: 10.1007/978-3-642-15534-5_17.

13. **Castillo, O. (2012).** Type-2 fuzzy logic in intelligent control applications, pp.1–188. DOI: 10.1007/978-3-642-24663-0.

14. **Castillo O., Melin P. (2001).** Simulation and forecasting complex economic time series using neural networks and fuzzy logic. International Joint Conference on Neural Networks, Vol. 3, pp. 1805–1810. DOI: 10.1109/IJCNN.2001.938436.

15. **Castillo O., Melin P. (2008).** Type-2 fuzzy systems. Type-2 Fuzzy logic Theory and Application, pp. 30–43. DOI: 10.1007/978-3-540-76284-3.

16. **Cowpertwait, P. S., Metcalfe, A. V. (2009).** Introductory time series with R. Springer Science and Business Media, pp. 2–5. DOI: 10.1007/978-0-387-88698-5.

17. **Davey, N., Hunt, S., Frank, R. (1999).** Time series prediction and neural networks. University of Hertfordshire, Hatfield.

18. **Euro Database (2021).** Futuros petróleo crudo WTI - Jul 2024 (CLN4). mx.investing.com/commodities/crude-oil.

19. **Escudero, P., Alcocer, W., Paredes, J. (2021).** Recurrent neural networks and arima models for euro/dollar exchange rate forecasting. Applied Sciences, Vol. 11, No. 12, pp. 5658. DOI: 10.3390/app11125658.

20. **Fathi, M., Kashani, M. H., Jameii, S. M., Mahdipour, E. (2021).** Big data analytics in

weather forecasting: a systematic review. Archives of Computational Methods in Engineering, Vol. 29, No. 2, pp. 1247–1275. DOI: 10.1007/s11831-021-09616-4.

21. **Nakip, M., Guzelis, C., Yildiz, O. (2021).** Recurrent trend predictive neural network for multi-sensor fire detection. IEEE Access, Vol. 9, pp. 84204–84216. DOI: 10.1109/access.2021.3087736.
22. **Simões-Hoffmann, L. F., Parquet-Bizarria, F. C., Parquet-Bizarria, J. W. (2020).** Detection of liner surface defects in solid rocket motors using multilayer perceptron neural networks. Polymer Testing, Vol. 88, pp. 106559. DOI: 10.1016/j.polymertesting.2020.106559.
23. **Ismail, F. H., Houssein, E. H., Hassanien, A. E. (2018).** Chaotic bird swarm optimization algorithm. Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2018, pp. 294–303. DOI: 10.1007/978-3-319-99010-1_27.
24. **Kennedy, J., Eberhard, R. (1975).** Particle swarm optimization. Proceedings of ICCN95 International Conference on Neuronal Networks, Vol. 4, pp. 1942–1948. DOI: 10.1109/ICNN.1995.488968.
25. **Connor, J., Martin, R., Atlas, L. (1994).** Recurrent neural networks and robust time series prediction. IEEE Transactions on Neural Networks, Vol. 5, No. 2, pp. 240–254. DOI: 10.1109/72.279188.
26. **Karnik, N. N., Mendel, J. M. (1998).** Introduction to type-2 fuzzy logic systems. IEEE International Conference on Fuzzy Systems Proceedings, Vol. 2. pp. 915–920. DOI: 10.1109/FUZZY.1998.686240.
27. **Kennedy, J., Eberhart, R. C., Shi, Y. H. (2001).** Swarm intelligence Algorithm Burlington, MK: Morgan Kaufmann Publishers.
28. **Koops, M. A., Giraldeau, L. A. (1996).** Producer–scrounger foraging games in starlings: A test of rate-maximizing and risk-sensitive models. Animal Behaviour, Vol. 51, No. 4, pp. 773–783. DOI: 10.1006/anbe.1996. 0082.
29. **Krausz, R. R. (2013).** Living in groups. Transactional Analysis Journal, Vol. 43, No. 4, pp. 366–374. DOI: 10.1177/0362153713519414.
30. **Lima, S. L., Dill, L. M. (1990).** Behavioral decisions made under the risk of predation: a review and prospectus. Canadian Journal of Zoology, Vol. 68, No. 4, pp. 619–640. DOI: 10.1139/z90-092.
31. **Tan, W. W., Chua, T. W. (2007).** Uncertain rule-based fuzzy logic systems: Introduction and new directions. IEEE Computational Intelligence Magazine, Vol. 2, No. 1, pp. 72–73. DOI: 10.1109/mci.2007.357196.
32. **Meng, X. B., Gao, X. Z., Lu, L., Liu, Y., Zhang, H. (2016).** A new bio-inspired optimisation algorithm: Bird swarm algorithm. Journal of Experimental and Theoretical Artificial Intelligence, Vol. 28, No. 4, pp. 673–687. DOI: 10.1080/0952813x.2015.1042530.
33. **Han, M., Xi, J., Xu, S., Yin, F. L. (2004).** Prediction of chaotic time series based on the recurrent predictor neural network. IEEE Transactions on Signal Processing, Vol. 52, No. 12, pp. 3409–3416. DOI: 10.1109/tsp.2004.837418.
34. **Tang, P., Zhang, Q., Zhang, X. (2023).** A recurrent neural network based generative adversarial network for long multivariate time series forecasting. Proceedings of the 2023 ACM International Conference on Multimedia Retrieval, pp. 181–189. DOI: 10.1145/3591106.3592306.
35. **Pulido, M., Melin, P., Mendoza, O. (2018).** Optimization of ensemble neural networks with type-1 and interval type-2 fuzzy integration for forecasting the Taiwan stock exchange. In: Gaweda, A., Kacprzyk, J., Rutkowski, L., Yen, G. (eds) Advances in Data Analysis with Computational Intelligence Methods. Studies

in Computational Intelligence, Springer, Cham, Vol 738. DOI: 10.1007/978-3-319-67946-4_6.

36. **Pulliam, H. R., Pyke, G. H., Caraco, T. (1982).** The scanning behavior of juncos: A game-theoretical approach. Journal of Theoretical Biology, Vol. 95, No. 1, pp. 89–103. DOI: 10.1016/0022-5193(82)90289-2.

37. **Pulliam, H. R (1973).** On the advantages of flocking. Journal of Theoretical Biology, Vol. 38, No. 2, pp. 419–422. DOI: 10.1016/0022-5193(73)90184-7.

38. **Mikolov, T., Kombrink, S., Burget, L., Černocký, J., Khudanpur, S. (2011).** Extensions of recurrent neural network language model. IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 5528–5531. DOI: 10.1109/icassp.2011.5947611.

39. **Petneházi, G. (2019).** Recurrent neural networks for time series forecasting. DOI: 10.48550/ARXIV.1901.00069.

40. **Chandra, R., Zhang, M. (2012).** Cooperative coevolution of Elman recurrent neural networks for chaotic time series prediction. Neurocomputing, Vol. 86, pp. 116–123. DOI: 10.1016/j.neucom.2012.01.014.

41. **Piccialli, F., Giampaolo, F., Prezioso, E., Camacho, D., Acampora, G. (2021).** Artificial intelligence and healthcare: Forecasting of medical bookings through multi-source time-series fusion. Information Fusion, Vol. 74, pp. 1–16. DOI: 10.1016/j.inffus.2021.03.004.

42. **Pulido, M., Melin, P. (2021).** Ensemble recurrent neural networks for complex time series prediction with integration methods. In: Castillo, O., Melin, P. (eds) Fuzzy Logic Hybrid Extensions of Neural and Optimization Algorithms: Theory and Applications. Studies in Computational Intelligence, Springer, Cham. Vol. 940, pp. 71–83. DOI: 10.1007/978-3-030-68776-2_4.

43. **Pulido, M., Melin, P. (2022).** Ensemble recurrent neural network using genetic algorithm applied in times series prediction. Computación y Sistemas, Vol. 26, No. 2, pp. 683–700. DOI: 10.13053/cys-26-2-4251.

44. **D'Urso, P., Giovanni, L., Massari, R. (2019).** Trimmed fuzzy clustering of financial time series based on dynamic time warping. Annals of Operations Research, Vol. 299, No. 1, pp. 1379–1395. DOI: 10.1007/s10479-019-03284-1.

45. **Sherstinsky, A. (2020).** Fundamentals of recurrent neural network (RNN) and long short term memory (LSTM) network. Physica D: Nonlinear Phenomena, Vol. 404, pp. 132306. DOI: 10.1016/j.physd.2019.132306.

46. **Saha, S., Raghava, G. P. S. (2006).** Prediction of continuous B-cell epitopes in an antigen using recurrent neural network. Proteins: Structure, Function, and Bioinformatics, Vol. 65, No. 1, pp. 40–48. DOI: 10.1002/prot.21078.

47. **Walid, A., Alamsyah, I. U. (2017).** Recurrent neural network for forecasting time series with long memory pattern. Journal of Physics: Conference Series, Vol. 824, No. 1, pp. 012038. DOI: 10.1088/1742-6596/824/1/012038.

48. **Wang, X., Wang, Y., Peng, J., Zhang, Z., Tang, X. (2022).** A hybrid framework for multivariate long-sequence time series forecasting. Applied Intelligence, Vol. 53, No. 11, pp. 13549–13568. DOI: 10.1007/s10489-022-04110-1.

49. **Wang, Y., Fang, S., Zhang, C., Xiang, S., Pan, C. (2022).** TVGCN: Time-variant graph convolutional network for traffic forecasting. Neurocomputing, Vol. 471, pp. 118–129. DOI: 10.1016/j.neucom.2021.11.006.

50. **Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., Cottrell, G. (2017).** A dual-stage attention-based recurrent neural network for time series prediction. DOI: 10.48550/arXiv.1704.02971.

51. Zhang, J., Man, K. F. (1998). Time series prediction using recurrent neural network in multidimension embedding phase space. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Vol. 2, pp. 1868–1873. DOI: 10.1109/ICSMC.1998.728168.

52. Zhang, J. S., Xiao, X. C. (2000). Predicting chaotic time series using recurrent neural network. Chinese Physics Letters, Vol. 17, No. 2, pp. 88–90. DOI: 10.1088/0256-307x/17/2/004.

53. Amador-Angulo, L., Ochoa, P., Peraza, C., Castillo, O. (2023). Fuzzy dynamic adaptation of an artificial fish swarm algorithm for the optimization of benchmark functions. In: Castillo, O., Melin, P. (eds) Hybrid Intelligent Systems Based on Extensions of Fuzzy Logic, Neural Networks and Metaheuristics. Studies in Computational Intelligence, Springer, Cham, Vol. 1096, pp. 99–114. DOI: 10.1007/978-3-031-28999-6_6.

54. Castillo, O., Lizárraga, E., Soria, J., Melin, P., Valdez, F. (2015). New approach using ant colony optimization with ant set partition for fuzzy control design applied to the ball and beam system. Information Sciences, Vol. 294, pp. 203–215. DOI: 10.1016/j.ins.2014.09.040.

55. Amador-Angulo, L., Mendoza, O., Castro, J., Rodríguez-Díaz, A., Melin, P., Castillo, O. (2016). Fuzzy sets in dynamic adaptation of parameters of a bee colony optimization for controlling the trajectory of an autonomous mobile robot. Sensors, Vol. 16, No. 9, pp. 1458. DOI: 10.3390/s16091458.

56. Sánchez, D., Melin, P., Castillo, O. (2017). A grey wolf optimizer for modular granular neural networks for human recognition. Computational Intelligence and Neuroscience, Vol. 2017, pp. 1–26. DOI: 10.1155/2017/4180510.

57. González, B., Valdez, F., Melin, P., Prado-Arechiga, G. (2015). Fuzzy logic in the gravitational search algorithm for the optimization of modular neural networks in pattern recognition. Expert Systems with Applications, Vol. 42, No. 14, pp. 5839–5847. DOI: 10.1016/j.eswa.2015.03.034.

58. Tai, K., El-Sayed, A. R., Biglarbegian, M., Gonzalez, C., Castillo, O., Mahmud, S. (2016). Review of recent type-2 fuzzy controller applications. Algorithms, Vol. 9, No. 2, pp. 39. DOI: 10.3390/a9020039.

59. Ontiveros, E., Melin, P., Castillo, O. (2020). Comparative study of interval type-2 and general type-2 fuzzy systems in medical diagnosis. Information Sciences, Vol. 525, pp. 37–53. DOI: 10.1016/j.ins.2020.03.059.

60. Melin, P., Castillo, O. (2004). A new method for adaptive control of non-linear plants using type-2 fuzzy logic and neural networks. International Journal of General Systems, Vol. 33, No. 2–3, pp. 289–304. DOI: 10.1080/03081070310001633608.

61. Moreno, J. E., Sanchez, M. A., Mendoza, O., Rodríguez-Díaz, A., Castillo, O., Melin, P., Castro, J. R. (2020). Design of an interval type-2 fuzzy model with justifiable uncertainty. Information Sciences, Vol. 513, pp. 206–221. DOI: 10.1016/j.ins.2019.10.042.

Article received on 26/12/2023; accepted on 07/05/2024.
**Corresponding author is Patricia Melin.*

ISSN 2007-9737

Comparative Study of Gorilla Troops Optimizerand Stochastic Fractal Search withFuzzy Dynamic Parameter Adaptation

Marylu L. Lagunes*, Ivette Miramontes, Oscar Castillo, Patricia Melin

Tijuana Institute of Technology, TecNM,
Mexico

marylu.lara@tectijuana.edu.mx, cynthiaivette84@gmail.com, {ocastillo, pmelin}@tectijuana.mx

Abstract. Metaheuristics has a very important role today in solving optimization problems; the vast majority of these methods are based on principles that imitate natural processes to achieve their results. The objective of this research is the analysis of the adaptability and stability of two bio-inspired methods, proposing a comparison between two optimization algorithms to evaluate and compare the performance and effectiveness of the algorithms in different optimization problems, the first, inspired by the social behavior of gorillas, which is called Artificial Gorilla Troops Optimizer (GTO), which is mathematically formulated to achieve exploration and exploitation in a given search space. The second algorithm is the one nature-inspired by imitating fractal behavior, known as Stochastic Fractal Search (SFS), where each of the particles moves stochastically until the objective function is found. By comparing both methods using benchmark functions, in this case CEC'2017 functions and performing the corresponding statistical analysis, we can conclude that with the GTO method, we obtained better results, since they are closer to the global optimum of the functions in comparison. with the SFS algorithm.

Keywords. Bio-inspired algorithms, fuzzy logic, optimization, CEC'2017 benchmark functions.

1 Introduction

Bio-inspired algorithms are those methods that try to imitate the behavior of the natural evolution of species in the search for mates and food. These forms of behavior are adapted to provide solutions to real problems in different algorithms.

Among the widely recognized methods are evolutionary algorithms, such as genetic algorithms, which are based on the evolution of species. This method has been considered to solve an endless number of optimization problems as described by the following authors [1, 2, 3, 4, 5, 6, 7]. Its versatility is evident in different contexts, such as in the development of computational algorithms capable of solving various problems in different fields, including pattern recognition, among others.

In the soft computing area, they are used to adjust membership functions in fuzzy controllers which help improve the performance of said systems. Bio-inspired optimization algorithms are characterized by being adaptive and non-deterministic, of which we find different applications today, for example, the design of communications antennas, and military tactics in airplanes [8, 9].

There is a wide variety of bio-inspired algorithms used for optimization, we will start by mentioning the Firefly Algorithm (FA) algorithm which is based on the inspiration of the behavior of flickering fireflies where each of the fireflies emits light or glow to find a partner and food.

This method is governed by 3 main rules and is widely used in the area of soft computing, in the following cases[10, 11, 12, 13, 14]. It describes how fireflies generate different values of possible solutions, for the optimization of fuzzy controllers that manage the behavior of an autonomous robot, in the area of artificial neural networks we can also find the use of bio-inspired algorithms as described in [15, 16, 17, 18, 19, 20], in addition to the mentioned methods, currently there are a wide variety of bio-inspired algorithms for optimization, the choice of them will depend on the problem to be solved, here we cite some[21, 22, 23, 24, 25].

The main objective of this research lies in evaluating and contrasting the performance of two algorithms, the bio-inspired Artificial Gorilla Troops

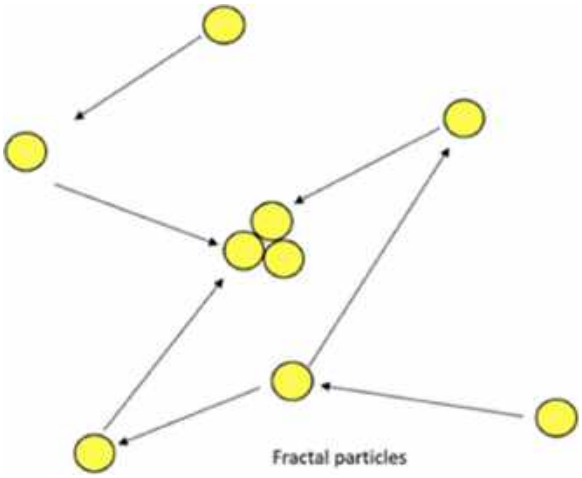


Fig. 1. Fractal particles movement

Optimizer (GTO) [26] and nature-inspired Stochastic Fractal Search (SFS) [27] using mathematical functions CEC 2017 [28].

The aim is to determine its efficiency, adaptability, stability, and effectiveness in various optimization problems. This comparison between algorithms intends to identify which of them is most effective in solving specific problems.

By analyzing their performance in specific cases, the aim is to discover the strengths and weaknesses of each algorithm, which would allow them to be adjusted and improve their applicability in more complex or specialized problems.

This comparative evaluation also contributes to a better understanding of algorithm approaches and their applicability in different problem domains. It is motivated by the search for the optimal solution in specific problems, as well as the dynamic adaptation capacity of the method, through the application of fuzzy logic to various variables to find the best solution.

The adaptability analysis was carried out by applying a fuzzy system, which made it possible to dynamically adjust key variables in each method and observe their behavior in the optimization of mathematical functions, particularly those of CEC 2017.

Fuzzy logic offers a mathematical framework which models non-linear functions, transforming inputs into outputs based on approximate reasoning. In this context, it was used to dynamically adjust the parameters of the compared methods, selecting said variables based on the design of the method and the needs posed by the problem in question.

This article is distributed as follows: Section 2 provides a review of the literature on nature-inspired metaheuristic algorithms used in this comparison and fuzzy logic. Section 3 describes the development of the comparison using fuzzy logic applied to CEC 2017 mathematical functions.

Section 4 observes the results obtained by the algorithms, showing the performance in tables and graphs. Section 5 deals with the discussion where the different points of view are presented according to the results obtained and Section 6 describes the conclusions and future work.

2 Literature Review

In this section, the bio-inspired approaches used in the research are detailed, along with the context related to fuzzy logic.

2.1 Artificial Gorilla Troops Optimizer (Algorithm 1)

Artificial Gorilla Troops Optimizer (GTO) [29] utilizes gorilla social behavior to enhance practical task performance.

It operates on a gorilla squad, where a primary model train specialized models for diverse tasks. These models benefit from knowledge transfer, expediting learning and enhancing outcomes.

The GTO method has demonstrated effectiveness in areas like natural language processing and computer vision, particularly in related responsibilities of varying complexity. Transferring knowledge from complex to specific tasks can enhance overall system performance.

However, it is crucial to note that GTO implementation may vary based on context and tasks, necessitating adjustments according to specific requirements and data [26, 30].

The exploratory phase initiates when a gorilla migrates, considering:

- When migrates to an unknown location.
- When migrates to a known location.
- Moving towards other gorillas.

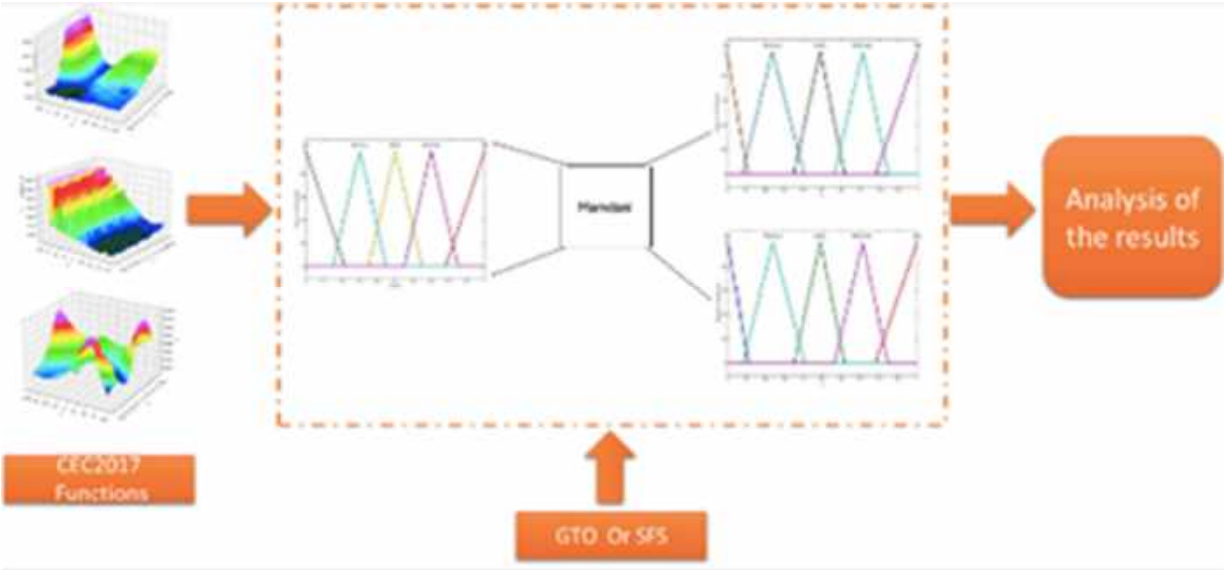


Fig. 2. Graphic description of the optimization process

The parameter P determines the migration mechanism to an unknown location. The initial mechanism activates when Rand < P. Contrariwise, if Rand ≥ 0.5, the movement shifts towards other gorillas. If Rand < 0.5, migration to a known location is favored. Eq. (1) summaries these three mechanisms of the exploratory phase:

$$GX(t+1) = \begin{cases} (UB - LB) \times r_1 + LB & rand < p \\ (r_1 - C) \times X_r(t) + L \times H & rand \geq 0.5 \\ X(i) - L \times \left(L \times \left(X(t) - GX_r(t)\right), + r_3 \times \left(X(t) - GX_r(t)\right)\right) & rand < p \end{cases} \quad (1)$$

where, $GX(t+1)$ denotes the individual next iteration candidate position vector at time (t), and $X(t)$ represents the current position vector. The values r_1, r_2, $r_3$and rand are random and range from 0 to 1, updated in each iteration.

The parameter p requires a predefined value before optimization, ranging from 0 to 1, determining the probability of selecting the migration mechanism to an unknown location. UB and LB indicate the upper and lower variable limits, respectively. X_r is randomly chosen from the gorilla population, while GX_r represents one of the candidate position vectors randomly selected, encompassing updated positions in each phase.

Finally, H is computed using equations (2), (4), and (5):

$$\mathrm{C} = \mathrm{F} * \left(1 - \frac{\mathrm{It}}{\mathrm{MaxIt}}\right), \quad (2)$$

$$F = \cos(2 \times r_4) + 1, \quad (3)$$

$$\mathrm{L} = \mathrm{C} \times \mathrm{l}. \quad (4)$$

In (2), t represents the current iteration value, $MaxIt$ stands for the total number of iterations for optimization, and F is determined by (3), wherein the cosine function cos and random values r_4 between 0 and 1 are employed, updating each iteration.

As per (2), initial optimization stages generate values with wide-ranging abrupt changes, which progressively narrow down towards the end.

In (4) computes L, with l being a random value ranging from -1 to 1, simulating the leadership role of the silverback gorilla. While the silverback gorilla may initially lack experience in making optimal decisions for food and group control, it gains stability and expertise over time.

The alterations in values are generated by (2) and (4). Additionally, in (1) calculates H using (5), and this, Z is determined through (6), where Z represents a random value within the problem is dimensions, ranging from -C to C:

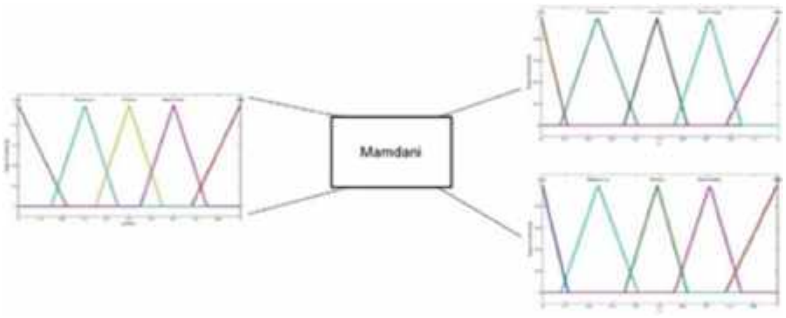


Fig. 3. GTO Triangular fuzzy inference system

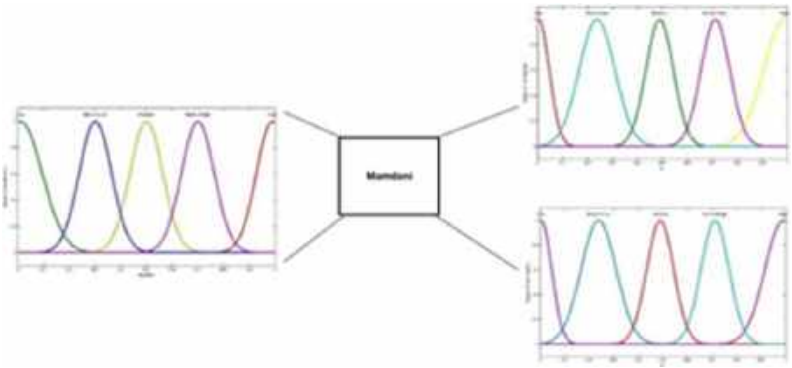


Fig. 4. GTO Gaussian fuzzy Inference system

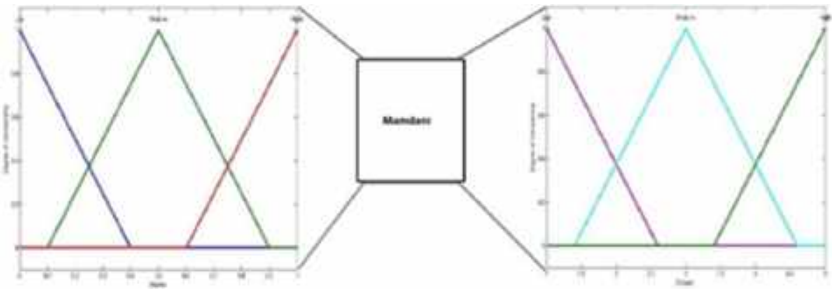


Fig. 5. SFS Triangular fuzzy inference system

$$H = Z \times X(t), \tag{5}$$

$$Z = [-C, C]. \tag{6}$$

In the exploitation phase of the method, two behaviors are employed: "Following the silverback" and "Competition for adult females."

The silverback gorilla leads the troop, makes decisions, leads movements and guides the gorillas to food sources, being responsible for the safety and well-being of the group.

All members adhere to its decisions. Nonetheless, the silverback gorilla can weaken, age, and eventually die, allowing the blackback of the group to assume leadership or other male gorillas to confront and dominate the silverback.

The decision between "Follow the Silverback" and "Competition for adult females" is made by the C value in (2). If C ≥ W, "Follow the silverback" is chosen; Otherwise, if C < W, "Competition for adult females" is chosen. W is a parameter established before the optimization process.

Table 1. CEC 2017 functions

No	Function	FI
	Unimodal Functions	
1	Shifted and Rotated Bent Cigar Function	100
2	Shifted and Rotated Sum of Different Power Function	200
3	Shifted and Rotated Zakharov Function	300
	Simple Multimodal Functions	
4	Shifted and Rotated Rosenbrock's Function	400
5	Shifted and Rotated Rastrigin's Function	500
6	Shifted and Rotated Expanded Scaffer's Function	600
7	Shifted and Rotated Lunacek Bi_RastriginFunction	700
8	Shifted and Rotated Non-Continuous Rastrigin's Function	800
9	Shifted and Rotated Levy Function	900
10	Shifted and Rotated Schwefel's Function	1000
	Hybrid functions	
11	Hybrid Function 1 (N = 3)	1100
12	Hybrid Function 2 (N = 3)	1200
13	Hybrid Function 3 (N = 3)	1300
14	Hybrid Function 4 (N = 4)	1400
15	Hybrid Function 5 (N = 4)	1500
16	Hybrid Function 6 (N = 4)	1600
17	Hybrid Function 6 (N = 5)	1700
18	Hybrid Function 6 (N = 5)	1800
19	Hybrid Function 6 (N = 5)	1900
20	Hybrid Function 6 (N = 6)	2000
	Composition Functions	
21	Composition Function 1 (N = 3)	2100
22	Composition Function 2 (N = 3)	2200
23	Composition Function 3 (N = 4)	2300
24	Composition Function 4 (N = 4)	2400
25	Composition Function 5 (N = 5)	2500
26	Composition Function 6 (N = 3)	2600
27	Composition Function 7 (N = 6)	2700
28	Composition Function 8 (N = 3)	2800
29	Composition Function 9 (N = 3)	2900
30	Composition Function 10 (N = 3)	3000

2.1.1 Following the Silverback

With the newly formed group, the silverback gorilla is young and in good health, like the other males in the group that faithfully follow him.

They all obey the silverback's orders to go to various areas in search of food sources. Additionally, members can influence the movement of the group. This approach is chosen when $C \geq W$. Eq. (7) is used to replicate this behavior:

$$GX(t+1) = L \times M \times (X(t) - rX_{siverback}) + X(t), \quad (7)$$

$$M = \left(\left| \sum_{i=1}^{N} GX(t) \right|^{g} \right)^{\frac{1}{g}}, \quad (8)$$

$$g = 2^{L}. \quad (9)$$

In (7), $X(t)$ represents the position vector of the gorilla, while X silverback is the position vector of the silverback gorilla. Furthermore, L is calculated using (4) and M using (8) where $GXi(t)$ shows the vector position of each candidate gorilla at iteration t. N denotes the total number of gorillas. g is also estimated using (9), and where L is also calculated using (4).

2.2 Stochastic Fractal Search (SFS) (Algorithm 2)

Fractals are formed by particles that move randomly, joining together to form a uniform pattern, each of the particles is added until they form a figure, the SFS has two important processes with which the method governs. The first process consists of particles diffusing near their position to comply with the intensification property, to find the global optimum and to avoid the local minimum.

This process is known as diffusion. The next consists of updating the position of a particle based on the position of the other particles in the group, the best particle produced is considered and the others will be discarded, this process also leads us to the diversification and Gaussian distribution for the random walk of particles and their growth in the diffusion process. The equations used in each of the aforementioned processes are explained below:

Table 2. GTO Triangular membership function

Triangular Membership Function					
Function	**F1**	**F2**	**F3**	**F4**	**F5**
Best	2.01×10^{4}	7.02×10^{15}	8.90×10^{3}	4.77×10^{2}	6.55×10^{2}
Worst	7.36×10^{6}	7.08×10^{26}	3.01×10^{4}	6.01×10^{2}	8.23×10^{2}
Average	9.56×10^{5}	3.91×10^{25}	1.88×10^{4}	5.27×10^{2}	7.74×10^{2}
STD	1.50×10^{6}	1.49×10^{26}	5.18×10^{3}	2.99×10^{1}	4.66×10^{1}
Function	**F6**	**F7**	**F8**	**F9**	**F10**
Best	6.34×10^{2}	9.54×10^{2}	9.03×10^{2}	3.63×10^{3}	4.35×10^{3}
Worst	6.66×10^{2}	1.35×10^{3}	1.03×10^{3}	5.87×10^{3}	9.28×10^{3}
Average	6.56×10^{2}	1.19×10^{3}	9.76×10^{2}	5.09×10^{3}	6.12×10^{3}
STD	8.59×10^{0}	1.02×10^{2}	3.05×10^{1}	5.16×10^{2}	1.37×10^{3}
Function	**F11**	**F12**	**F13**	**F14**	**F15**
Best	1.17×10^{3}	1.13×10^{5}	4.05×10^{3}	1.82×10^{3}	1.83×10^{3}
Worst	1.38×10^{3}	6.47×10^{6}	1.64×10^{5}	5.04×10^{4}	3.82×10^{4}
Average	1.25×10^{3}	2.24×10^{6}	2.49×10^{4}	1.16×10^{4}	1.16×10^{4}
STD	4.68×10^{1}	1.33×10^{6}	3.27×10^{4}	1.27×10^{4}	9.89×10^{3}
Function	**F16**	**F7**	**F18**	**F19**	**F20**
Best	1.92×10^{3}	1.89×10^{3}	3.15×10^{4}	2.03×10^{3}	2.31×10^{3}
Worst	3.52×10^{3}	2.93×10^{3}	3.15×10^{6}	1.88×10^{4}	3.20×10^{3}
Average	2.79×10^{3}	2.38×10^{3}	2.04×10^{5}	6.76×10^{3}	2.72×10^{3}
STD	3.76×10^{2}	2.89×10^{2}	2.57×10^{5}	5.16×10^{3}	2.15×10^{2}
Function	**F21**	**F22**	**F23**	**F24**	**F25**
Best	2.22×10^{3}	2.31×10^{3}	2.80×10^{3}	2.92×10^{3}	2.89×10^{3}
Worst	2.54×10^{3}	9.76×10^{3}	3.15×10^{3}	3.47×10^{3}	2.95×10^{3}
Average	2.44×10^{3}	4.07×10^{3}	2.91×10^{3}	3.07×10^{3}	2.92×10^{3}
STD	5.76×10^{1}	2.62×10^{3}	8.17×10^{1}	1.24×10^{2}	1.57×10^{1}
Function	**F26**	**F27**	**F28**	**F29**	**F30**
Best	3.21×10^{3}	3.24×10^{3}	3.21×10^{3}	2.81×10^{3}	9.09×10^{3}
Worst	8.31×10^{3}	3.71×10^{3}	3.33×10^{3}	5.96×10^{3}	7.05×10^{5}
Average	6.14×10^{3}	3.39×10^{3}	3.28×10^{3}	6.42×10^{3}	5.59×10^{4}
STD	1.45×10^{3}	1.14×10^{2}	2.82×10^{1}	4.46×10^{2}	1.24×10^{5}

$$P = LB + \varepsilon * (UP - LB). \tag{10}$$

The particle population P is randomly generated based on the problem constraints after setting the lower (LB) and the upper (UB) limits, where ε is a random number in the range [0,1].

The diffusion process (exploitation in fractal search) is expressed as follows:

$$GW_1 = \text{Gaussian}(\mu_{BP}, \sigma) + \left(\varepsilon * BP - \varepsilon' * P_i\right), \tag{11}$$

$$GW_2 = \text{Gaussian}\,(\mu_P, \sigma), \tag{12}$$

where ε, ε' represent random numbers in the range [0,1], BP is the best position of the point, i-th point in the group is represented by P_i and Gaussians a function that generates a random number from the normal distribution with a mean μ parameter and a standard deviation parameter σ:

$$\sigma = \frac{\log g}{g} \times |P_i - BP| \tag{13}$$

Table 3. GTO Gaussian membership function

Gaussian Membership Function					
Function	**F1**	**F2**	**F3**	**F4**	**F5**
Best	3.75×10^{4}	2.43×10^{15}	8.72×10^{3}	4.75×10^{2}	6.18×10^{2}
Worst	6.36×10^{6}	3.22×10^{28}	3.54×10^{4}	6.35×10^{2}	8.21×10^{2}
Average	1.20×10^{6}	1.14×10^{27}	1.92×10^{4}	5.22×10^{2}	7.42×10^{2}
STD	1.45×10^{6}	5.87×10^{27}	7.08×10^{3}	3.09×10^{1}	5.65×10^{1}
Function	**F6**	**F7**	**F8**	**F9**	**F10**
Best	1.06×10^{3}	9.19×10^{2}	3.03×10^{3}	4.73×10^{3}	1.06×10^{3}
Worst	1.37×10^{3}	1.03×10^{3}	5.78×10^{3}	8.20×10^{3}	1.37×10^{3}
Average	1.22×10^{3}	9.80×10^{2}	5.12×10^{3}	6.45×10^{3}	1.22×10^{3}
STD	8.85×10^{1}	3.16×10^{1}	5.51×10^{2}	8.67×10^{2}	8.85×10^{1}
Function	**F11**	**F12**	**F13**	**F14**	**F15**
Best	1.18×10^{3}	2.10×10^{5}	3.74×10^{3}	1.65×10^{3}	2.02×10^{3}
Worst	1.40×10^{3}	6.99×10^{6}	4.46×10^{6}	3.62×10^{4}	3.12×10^{4}
Average	1.26×10^{3}	1.91×10^{6}	1.93×10^{5}	9.51×10^{3}	8.03×10^{3}
STD	4.85×10^{1}	1.61×10^{6}	8.14×10^{5}	9.76×10^{3}	6.59×10^{3}
Function	**F16**	**F7**	**F18**	**F19**	**F20**
Best	2.32×10^{3}	1.82×10^{3}	2.72×10^{4}	2.08×10^{3}	2.35×10^{3}
Worst	3.80×10^{3}	2.84×10^{3}	1.02×10^{6}	2.03×10^{4}	3.25×10^{3}
Average	2.98×10^{3}	2.37×10^{3}	1.68×10^{5}	7.09×10^{3}	2.63×10^{3}
STD	3.76×10^{2}	2.74×10^{2}	1.87×10^{5}	4.63×10^{3}	2.13×10^{2}
Function	**F21**	**F22**	**F23**	**F24**	**F25**
Best	2.41×10^{3}	2.30×10^{3}	2.79×10^{3}	2.93×10^{3}	2.89×10^{3}
Worst	2.56×10^{3}	1.02×10^{4}	3.08×10^{3}	4.06×10^{3}	2.96×10^{3}
Average	2.46×10^{3}	4.97×10^{3}	2.90×10^{3}	3.10×10^{3}	2.92×10^{3}
STD	4.37×10^{1}	2.99×10^{3}	7.31×10^{1}	2.11×10^{2}	1.94×10^{1}
Function	**F26**	**F27**	**F28**	**F29**	**F30**
Best	2.93×10^{3}	3.21×10^{3}	3.24×10^{3}	3.78×10^{3}	8.02×10^{3}
Worst	8.17×10^{3}	3.77×10^{3}	3.37×10^{3}	5.13×10^{3}	2.43×10^{7}
Average	5.76×10^{3}	3.35×10^{3}	3.29×10^{3}	4.42×10^{3}	8.60×10^{5}
STD	1.45×10^{3}	1.17×10^{2}	2.77×10^{1}	3.50×10^{2}	4.43×10^{6}

where $\log g \,/\, g$ tends to a zero value as the generation number g increases. The update process (fractal search exploration) is:

$$\mathrm{Pa_i} = \frac{\mathrm{rank(P_i)}}{\mathrm{N}}, \tag{14}$$

where *N* is the number of particles in the group and Pa_i is the estimated probability value for a particle whose rank among the others is given by the "rank" function.

The particles are classified according to their fitness value and then a probability value is given to each particle *i*:

$$P_i'(j) = P_x(j) - \varepsilon * \left(P_y(j) - P_i(j)\right), \tag{15}$$

where the augmented component is represented by $P_i'(j)$, and P_x, P_y are different points randomly chosen from the group. P_i' replacements P_i if it has a better fitness value:

$$P_i' = P_i - \varepsilon * (P_x - BP) | \varepsilon' \leq 0.5 , \tag{16}$$

Table 4. SFS Gaussian membership function

Gaussian Membership Function					
Function	**F1**	**F2**	**F3**	**F4**	**F5**
Best	4.97×10^{4}	2.63×10^{36}	5.77×10^{4}	4.87×10^{2}	7.60×10^{2}
Worst	3.70×10^{5}	3.87×10^{40}	6.17×10^{4}	6.56×10^{2}	8.46×10^{2}
Average	1.58×10^{5}	2.07×10^{39}	5.54×10^{4}	5.56×10^{2}	7.84×10^{2}
STD	7.35×10^{4}	7.57×10^{39}	8.70×10^{3}	4.23×10^{1}	2.91×10^{1}
Function	**F6**	**F7**	**F8**	**F9**	**F10**
Best	6.00×10^{2}	9.96×10^{2}	1.02×10^{3}	9.47×10^{2}	1.07×10^{4}
Worst	6.01×10^{2}	1.03×10^{3}	1.11×10^{3}	1.15×10^{3}	1.18×10^{4}
Average	6.01×10^{2}	1.06×10^{3}	1.08×10^{3}	1.10×10^{3}	1.12×10^{4}
STD	2.05×10^{-1}	2.58×10^{1}	3.09×10^{1}	1.24×10^{2}	5.78×10^{2}
Function	**F11**	**F12**	**F13**	**F14**	**F15**
Best	1.32×10^{3}	2.12×10^{6}	2.26×10^{3}	1.93×10^{3}	3.88×10^{3}
Worst	1.42×10^{3}	9.29×10^{6}	1.00×10^{5}	2.00×10^{3}	9.75×10^{3}
Average	1.36×10^{3}	3.77×10^{6}	2.19×10^{4}	1.91×10^{3}	4.64×10^{3}
STD	2.97×10^{1}	1.50×10^{6}	1.93×10^{4}	1.04×10^{2}	1.39×10^{3}
Function	**F16**	**F7**	**F18**	**F19**	**F20**
Best	3.07×10^{3}	2.91×10^{3}	2.57×10^{4}	3.46×10^{3}	2.77×10^{3}
Worst	4.01×10^{3}	3.42×10^{3}	1.26×10^{5}	1.56×10^{4}	3.03×10^{3}
Average	3.65×10^{3}	3.15×10^{3}	5.32×10^{4}	6.00×10^{3}	3.15×10^{3}
STD	3.02×10^{2}	2.20×10^{2}	2.20×10^{4}	2.49×10^{3}	2.23×10^{2}
Function	**F21**	**F22**	**F23**	**F24**	**F25**
Best	2.54×10^{3}	2.31×10^{3}	1.32×10^{4}	3.09×10^{3}	3.03×10^{3}
Worst	2.58×10^{3}	1.32×10^{4}	3.05×10^{3}	3.22×10^{3}	3.11×10^{3}
Average	2.57×10^{3}	1.12×10^{4}	3.00×10^{3}	3.17×10^{3}	3.06×10^{3}
STD	2.95×10^{1}	4.16×10^{3}	3.22×10^{1}	3.78×10^{1}	2.19×10^{1}
Function	**F26**	**F27**	**F28**	**F29**	**F30**
Best	5.82×10^{3}	3.36×10^{3}	3.32×10^{3}	3.60×10^{3}	2.93×10^{6}
Worst	7.01×10^{3}	3.38×10^{3}	3.29×10^{3}	4.42×10^{3}	5.08×10^{6}
Average	6.39×10^{3}	3.43×10^{3}	3.35×10^{3}	4.18×10^{3}	3.95×10^{6}
STD	3.65×10^{2}	3.85×10^{1}	2.92×10^{1}	2.42×10^{2}	7.45×10^{5}

$$P_i' = P_i - \varepsilon * \left(P_x - P_y\right) | \text{ otherwise.} \quad (17)$$

At the end of the first update process, the second one begins by ranking all the resulting points once more based on Eqs. (16) and (17).

As before, if Pa_i is less than a random number ε, the current point, P_i is modified by using the previous equations, where the $x$and y indices must be different. Of course, the new pointP_i' is replaced by P_i if it is better than P_i.

2.3 Fuzzy Logic

Fuzzy logic developed by Lotfi Zadeh [31] is a type of mathematical logic that allows for handling ambiguous or imprecise concepts. Unlike classical logic, which assumes that an object or a proposition is true or false in an exclusive manner, fuzzy logic allows us to represent the degree of membership or veracity of an object in a fuzzy set. In fuzzy logic, truth values can be any number

Table 5. Parameters used in Z-test for GTO VS SFS

Parameter of Z-test for GTO vs SFS	
Critical Value (Zc)	1.64
Confidence interval	95%
H_0	$\mu1 \geq \mu2$
H_a (Claim)	$\mu1 < \mu2$
Alpha	0.05

between 0 and 1, meaning that an object can have a degree of partial membership in a set rather than simply being true or false.

This allows modelling and representing situations in which uncertainty, imprecision or subjectivity are present. Fuzzy logic has been successfully applied in various fields such as artificial intelligence, system control, decision-making, pattern recognition and engineering, among others. Its applications are based on the ability to handle and reason with incomplete or uncertain information as described by the following authors.[32, 33, 34, 35].

A fuzzy inference system is a system that uses fuzzy logic to perform reasoning and decision-making based on uncertainty and imprecision of data. A fuzzy inference system consists of three main components:

1 The fuzzy knowledge base: it is where the rules that relate the input variables to the output variables are defined.

2 These rules are formulated in linguistic terms and are based on expert knowledge of the domain.

3 The fuzzy inference engine: it is responsible for combining the rules of the knowledge base and calculating the output in fuzzy terms. It uses different inference methods, such as the Mamdani method or the maximum method.

4 The fuzzy database: contains the input information necessary for the system to make inferences. The input variables are represented by fuzzy sets, which assign a degree of membership to each possible value.

The fuzzy inference system can be used in various fields, such as engineering, medicine, robotics, and control systems, among others. Its ability to handle imprecision and uncertainty makes it especially useful when available data is incomplete or ambiguous.[36, 37].

3 Comparison Analysis with Dynamic Parameter Adjustment in the Bio-Inspired Methods

The CEC 2017 mathematical functions are sets of known problems, which are used to evaluate optimization algorithms. These functions provide a common framework for comparing the performance of different algorithms [38, 39].

Table 1 shows the mathematical functions of the CEC 2017, which are classified as unimodal, multimodal, hybrid and composite functions. The number of functions is 30, each of which has a different global value.

These values are what the algorithms that undergo this type of testing must find since they are the way to evaluate the behavior and effectiveness of the methods. Next, the dynamic parameter adaptation process used to optimize the membership functions of the fuzzy systems used is described.

In the first instance, it begins with the analysis of the problem, in this case, it corresponds to dynamically adapting the values of the parameters of the membership functions to improve the performance of the method in the search for the global optimum for each mathematical function.

Afterwards, the construction of the fuzzy system continues, for which the optimization method is explored to know the variables to which said adjustment is applied. Once found, they are used as outputs of the fuzzy system, and the iteration is used at the input, so that, in each iteration (each time this occurs in the method), the dynamic adaptation to the chosen variables would be performed.

Figure 4 shows the fuzzy system used for the dynamic parameters adaptation in the GTO method. For this first case study, triangular membership functions are used to analyze its operation. Figure 5 shows the fuzzy system used for dynamic parameter adaptation also implemented in the GTO method.

In this second case study, Gaussian membership functions are used. It should be noted

Table 6. Z-test results

	Calculated z	Evidence
1	3,931	not significan
2	-1,498	not significan
3	-17,677	significan
4	-3,555	significan
5	-3.62	significan
6	46,611	not significan
7	9,507	not significan
8	-12,393	significan
9	38,986	not significan
10	-24,968	significan
11	-9,631	significan
12	-4.63	significan
13	1,151	not significan
14	4,265	not significan
15	2,757	not significan
16	-7,609	significan
17	-12,158	significan
18	3,339	not significan
19	1,136	not significan
20	-9,236	significan
21	-11,427	significan
22	-6,661	significan
23	-6,857	significan
24	-1,789	significan
25	-26.21	significan
26	-2,308	significan
27	-3,557	significan
28	-8,165	significan
29	3,089	not significan
30	-3,768	significan

that both fuzzy systems use the iterations as input and adjust the parameters of p and $\boldsymbol{W}$, as outputs.

Figure 6 shows the fuzzy system used for the dynamic parameters adaptation also implemented in the SFS method, using the iteration as input and the diffusion parameter as output. For this case study, triangular membership functions are used. The fuzzy rules are listed as follows:

1 If iteration is Low then p is High and w is MediumLow.

2 If iteration is MediumLow then p is Medium and w is Medium

3 If iteration is Medium then p is MediumLow and w is MediumHigh.

4 If iteration is MediumHigh then p is Low and w is High.

5 If iteration is High then p is Low and w is High.

To execute the methods, the following architecture was used: In the GTO alpha 0.1 algorithm, delta 0.1, agents 50, Iterations 500, dimensions 50 and for SFS agents 50, Iterations 500, dimensions 50.

4 Results

Table 2 presents the results derived from the use of a fuzzy system with triangular membership functions. This system was developed to improve the search for the global optimum in the CEC 2017 functions.

The first column describes the metrics used to analyze the results. It starts with the function evaluated, followed by the best result obtained for that function about the number of experiments established.

Subsequently, the worst result obtained under the same number of experiments is presented. Likewise, the average is offered as a measure of central tendency that indicates the center of the results within the statistical distribution.

Finally, the standard deviation is shown, which indicates the dispersion of the data in relation to the mean. In particular, in the functions *f4, f5, f6, f7* and *f8*, the values obtained were very close to the corresponding optima of those function.

Table 3 presents the results obtained by using the fuzzy system, this time developed with Gaussian membership functions. This variation was carried out to determine which of the systems offers better performance when searching for the values of mathematical functions.

The structure and content of the table are similar to those of Table 2, which details the metrics used to analyze the results, including the function evaluated, the best and worst result, the average and the standard deviation, giving the best results in the functions f4, f5, f6 and f8 approaching the global optimum.

This approach of employing Gaussian membership functions represents an additional

exploration to determine the effectiveness of the fuzzy system in optimizing mathematical functions. The results of the SFS method with dynamic parameter adaptation are detailed in Table 4.

This table shows the values corresponding to the best result obtained, the worst result, the average, and the standard deviation, presented in a format similar to the previous tables, to provide a complete view of the performance of the method.

Statistical test.

The Z parametric test is used to perform the statistical analysis, to compare the results obtained throughout the experimentation. Mathematically, the statistical test is expressed as:

$$Z = \frac{(\bar{X}_1 - \bar{X}_2) - (\mu_1 - \mu_2)}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}, \tag{18}$$

were, $\bar{x}_1 - \bar{x}_2$ it represents the difference between the sample means, and $\mu_1 - \mu_2$ denotes the difference between the population means:

$$\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}. \tag{19}$$

Represents the sum of the population standard deviations, and (n_1, n_2) represents the sample size. In the experiments carried out with the mathematical functions of CEC2017, where Type-1 fuzzy systems are used for the adaptation of dynamic parameters, the following hypotheses are established:

- Null hypothesis (Ho): The results provided by the GTO using dynamic parameter tuning with Gaussian membership functions are greater than or equal to those of the SFS method using dynamic parameter adaptation with Gaussian membership functions.
- Alternative hypothesis (Ha): The results provided by the GTO using dynamic parameter adjustment with Gaussian membership functions are lower than those obtained by the SFS method using dynamic parameter adaptation with Trapezoidal membership functions. Table 5 presents the statistical parameters used for this analysis:

The results of the Z test applied to the thirty CEC2017 functions are shown in Table 3. Columns 2 and 3 show the results of the GTO using Gaussian membership functions and their standard deviation, while columns 4 and 5 present the results of the SFS using Gaussian membership functions.

Column 6 describes the results of the Z test, and column 7 indicates whether there is (Y) or not (NS) significant evidence to reject the null hypothesis. It can be seen that in 19 of the 30 functions, there is evidence that supports the claim that the GTO method dynamic adjustment with Gaussian membership functions provides better results, therefore it can be said that the GTO method is capable of adapting to obtain the optimum of each function demonstrating its effectiveness when adjusted with Type-1 fuzzy logic.

In Table 6 you can see column 1 where the benchmark functions are shown, column 2 corresponds to the Z calculated which refers to the value obtained when performing a hypothesis test, this is used to evaluate statements about the mean of a population when the standard deviation is known.

The "critical z" is the critical value of the z statistic that is used in hypothesis testing to establish a boundary between the rejection region and the non-rejection region, to all functions $z = -1.64$. In hypothesis testing, the "calculated z" is usually compared to the "critical z" to make decisions about whether or not to reject the null hypothesis.

5 Discussion

The purpose of this study is to compare the adaptive capacity of the GTO and SFS methods to identify global optima in the CEC 2017 functions. Type-1 fuzzy logic was used for dynamically adapting the parameters of the membership functions, seeking to improve the performance of both methods.

The mathematical functions evaluated have varying degrees of complexity, resulting in different global optima. This diversity challenges the methods and demonstrates their performance and effectiveness in solving complex problems.

After meticulously analyzing each method, variables that required dynamic adjustments were identified. Fuzzy logic, widely used in solving optimization problems with bio-inspired algorithms, was crucial to determine which method was best suited and offered the best results.

When observing the results in Tables 2 and 3, it is noted that the results obtained by the GTO method, using triangular membership functions, and the fuzzy system using Gaussian functions, showed significant similarities.

This suggests that the difference between both systems was minimal, demonstrating a similar adaptability of the dynamically adjusted method in each iteration, which supports its effectiveness in performance with dynamic parameter adjustment.

Performing dynamic parameter adaptation with Type-1 fuzzy logic has been essential to improve the approximation of the methods to the optimal values of the functions, despite its complexity. However, the SFS algorithm, as detailed in Table 4, faced more difficulties in reaching the optimal values.

Their results were found to be further from the real values, indicating that their adaptability with dynamic parameter adaptation was not as effective as in the case of the GTO method.

6 Conclusions and Future Work

In this article, a comprehensive case study was conducted that compared two bioinspired methods to analyze their performance and adaptability. Benchmark mathematical functions from CEC 2017 were used as evaluation tools, which present different levels of complexity and, therefore, different global optima.

The results obtained were highly positive. The GTO method, which uses a fuzzy system with Gaussian membership functions and triangular membership functions, showed significant improvements in functions 4, 5, 6, 7 and 8, as detailed in tables 3 and 4. On the other hand, the FSF method demonstrated good results in functions 4, 5, 6, 7 and 9, as seen in Table 5.

The results obtained reflect the effectiveness of both methods to find the optimum in some of the evaluated functions. This comparative analysis provides a clear view of the relative performance of the algorithms in different contexts and lays the foundation for future research and improvements in the optimization of specific functions.

The evaluation of the FSF and GTO methods was carried out using Type-1 fuzzy logic, to adjust the parameters of the membership functions used in the fuzzy controller, this optimization was carried out to know the behavior of the methods when searching for the global of each function, where satisfactory results were obtained, although they can be improved using other intelligent computing techniques such as type 2 fuzzy logic, like in [40-42].Type-2 fuzzy logic will help us have a better insertion threshold and find better results.

It is intended to use different membership functions, in addition to adding more inputs to fuzzy systems to improve their performance, such as diversity.

Some type of hybridization between the methods can also be carried out since both have proven to be efficient for specific optimizations. In order to carry out some type of hybridization, it will be necessary to thoroughly analyze each part of the algorithms, especially those parameters that help their convergence and performance.

References

1. **Holland, J. H. (1967).** Genetic algorithms understand genetic algorithms. Surprise, Vol. 96, No. 1, pp. 12–15.
2. **Pal, S. K., Wang, P. P. (1996).** Genetic algorithms for pattern recognition. CRC Press.
3. **Wu, D., Tan, W. W. (2006).** Genetic learning and performance evaluation of interval Type-2 fuzzy logic controllers. Engineering Applications of Artificial Intelligence, Vol. 19, No. 8, pp. 829–841. DOI: 10.1016/j.engappai. 2005.12.011.
4. **Babaie-Kafaki, S., Ghanbari, R., Mahdavi-Amiri, N. (2016).** Hybridizations of genetic algorithms and neighborhood search metaheuristics for fuzzy bus terminal location problems. Applied Soft Computing, Vol. 46, pp. 220—229. DOI: 10.1016/j.asoc.2016.03.005.
5. **Luthra, J., Pal, S. K. (2011).** A hybrid firefly algorithm using genetic operators for the cryptanalysis of a monoalphabetic substitution

cipher. World Congress on Information and Communication Technologies, pp. 202—206. DOI: 10.1109/wict.2011.6141244.

6. **Han, S., Pedrycz, W., Han, C. (2005).** Nonlinear channel blind equalization using hybrid genetic algorithm with simulated annealing. Mathematical and Computer Modelling, Vol. 41, No. 6-7, pp. 697–709. DOI: 10.1016/j.mcm.2004.05.006.
7. **Wagner, C., Hagras, H. (2007)**. A genetic algorithm based architecture for evolving Type-2 fuzzy logic controllers for real world autonomous mobile robots. IEEE International Fuzzy Systems Conference, pp. 1–6. DOI: 10.1109/fuzzy.2007.4295364.
8. **Trrad, I., Smadi, T. A., Al_Wahshat, H. (2019).** Application of fuzzy logic to cognitive wireless communications. International Journal of Recent Technology and Engineering (IJRTE), Vol. 8, No. 3, pp. 2228–2234. DOI: 10.35940/ijrte.b2065.098319.
9. **Pratiwi, W., Sofwan, A., Setiawan, I. (2020).** Implementation of fuzzy logic method for automation of decision making of Boeing aircraft landing. IAES International Journal of Artificial Intelligence, Vol. 10, No. 3. DOI: 10.11 591/ijai.v10.i3.
10. **Yang, X. S., He, X. (2013).** Firefly algorithm: recent advances and applications. International Journal of Swarm Intelligence, Vol. 1, No. 1, pp. 36–50. DOI: 10.1504/ijsi. 2013.055801.
11. **Yang, X. (2009).** Firefly algorithm, Lévy flights and global optimization. Research and Development in Intelligent Systems XXVI, pp. 209–218. DOI: 10.1007/978-1-84882-983- 1_15.
12. **Lagunes, M. L., Castillo, O., Soria, J., Garcia, M., Valdez, F. (2018).** Optimization of granulation for fuzzy controllers of autonomous mobile robots using the firefly algorithm. Granular Computing, Vol. 4, No. 2, pp. 185–195. DOI: 10.1007/s41066-018-0121- 6.
13. **Lagunes, M. L., Castillo, O., Soria, J., Garcia, M., Valdez, F. (2018).** Optimization of granulation for fuzzy controllers of autonomous mobile robots using the firefly algorithm. Granular Computing, Vol. 4, No. 2, pp. 185–195. DOI: 10.1007/s41066-018-0121- 6.
14. **Lagunes, M. L., Castillo, O., Valdez, F., Soria, J., Melin, P. (2018)**. Parameter optimization for membership functions of type-2 fuzzy controllers for autonomous mobile robots using the firefly algorithm. Fuzzy Information Processing, pp. 569–579. DOI: 10.1007/978-3-319-95312-0_50.
15. **Miramontes, I., Guzman, J., Melin, P., Prado-Arechiga, G. (2018).** Optimal design of interval type-2 fuzzy heart rate level classification systems using the bird swarm algorithm. Algorithms, Vol. 11, No. 12, pp. 206. DOI: 10.3390/a11120206.
16. **Tarik, R., Adil, B. (2022).** Ant colony optimization algorithm and fuzzy logic for switched reluctance generator control. AIMS Energy, Vol. 10, No. 5, pp. 987–1004. DOI: 10.3934/energy.2022045.
17. **Mohamed, A., Saber, W., Elnahry, I., Hassanien, A. E. (2020).** Coyote optimization based on a fuzzy logic algorithm for energy-efficiency in wireless sensor networks. IEEE Access, Vol. 8, pp. 185816–185829. DOI: 10. 1109/access.2020.3029683.
18. **Bhoi, A. K., Mallick, P. K., Liu, C. M., Balas, V. E. (Eds.). (2021).** Bio-inspired neurocomputing. Berlin/Heidelberg, Germany: Springer, Vol. 310. DOI: 10.1007/978-981-15-5495-7.
19. **Miramontes, I., Melin, P. (2022).** Interval type-2 fuzzy approach for dynamic parameter adaptation in the bird swarm algorithm for the optimization of fuzzy medical classifier. Axioms, Vol. 11, No. 9, pp. 485. DOI: 10.3390/ axioms11090485.
20. **Miramontes, I., Melin, P. (2023).** Enhancing dynamic parameter adaptation in the bird swarm algorithm using general type-2 fuzzy analysis and mathematical functions. Axioms, Vol. 12, No. 9, pp. 834. DOI: 10.3390/ axioms12090834.
21. **Glover, F. (1990).** Tabu search: A tutorial. Interfaces, Vol. 20, No. 4, pp. 74–94. DOI: 10. 1287/inte.20.4.74.

22. **Dorigo, M., Stützle, T. (2003).** The ant colony optimization metaheuristic: algorithms, applications, and advances. Handbook of Metaheuristics, pp. 250–285. DOI: 10.1007/0-306-48056-5_9.
23. **Xu, Y., Fan, P., Yuan, L. (2013).** A simple and efficient artificial bee colony algorithm. Mathematical Problems in Engineering, Vol. 2013, pp. 1–9. DOI: 10.1155/2013/526315.
24. **Patwal, R. S., Narang, N., Garg, H. (2018).** A novel TVAC-PSO based mutation strategies algorithm for generation scheduling of pumped storage hydrothermal system incorporating solar units. Energy, Vol. 142, pp. 822–837. DOI: 10.1016/j.energy.2017.10.052.
25. **Brito, J., Martínez, F., Moreno, J., Verdegay, J. (2015).** An ACO hybrid metaheuristic for close–open vehicle routing problems with time windows and fuzzy constraints. Applied Soft Computing, Vol. 32, pp. 154–163. DOI: 10.1016/j.asoc.2015.03.026.
26. **Gomaa, I. H., Zaher, H., Ragaa-Saeid, N., Sayed, H. (2023).** A novel enhanced gorilla troops optimizer algorithm for global optimization problems. International Journal of Industrial Engineering & Production Research, Vol. 34, No. 1, pp. 1–9. DOI: 10.22068/ijiepr.34.1.3.
27. **Salimi, H. (2015).** Stochastic fractal search: A powerful metaheuristic algorithm. Knowledge-Based Systems, Vol. 75, pp. 1–18. DOI: 10.1016/j.knosys.2014.07.025.
28. **Awad, N., Ali, M., Liang, J., Qu, B., Suganthan, P., Definitions, P. (2016).** Evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization. Technical Report, pp. 5–8.
29. **Sayed, G. I., Hassanien, A. E. (2021).** A novel chaotic artificial gorilla troops optimizer and its application for fundus images segmentation. Lecture Notes on Data Engineering and Communications Technologies, pp. 318–329. DOI: 10.1007/978-3-030-89701-7_28.
30. **Abdollahzadeh, B., Gharehchopogh, F. S., Mirjalili, S. (2021).** Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. International Journal of Intelligent Systems, Vol. 36, No. 10, pp. 5887--5958. DOI: 10.1002/int.22535.
31. **Zadeh, L. (1988).** Fuzzy logic. Computer, Vol. 21, No. 4, pp. 83–93. DOI: 10.1109/2.53.
32. **Karnik, N., Mendel, J., Liang, Q. (1999).** Type-2 fuzzy logic systems. IEEE Transactions on Fuzzy Systems, Vol. 7, No. 6, pp. 643–658. DOI: 10.1109/91.811231.
33. **Schouten, N., Salman, M., Kheir, N. (2002).** Fuzzy logic control for parallel hybrid vehicles. IEEE Transactions on Control Systems Technology, Vol. 10, No. 3, pp. 460–468. DOI: 10.1109/87.998036.
34. **Liang, Q., Mendel, J. (2000).** Interval type-2 fuzzy logic systems: theory and design. IEEE Transactions on Fuzzy Systems, Vol. 8, No. 5, pp. 535–550. DOI: 10.1109/91.873577.
35. **Mendel, J. M., John, R. I., Liu, F. (2006).** Interval type-2 fuzzy logic systems made simple. IEEE Transactions on Fuzzy Systems, Vol. 14, No. 6, pp. 808–821. DOI: 10.1109/tfuzz.2006.879986.
36. **Zadeh, L. A. (2008).** Is there a need for fuzzy logic? Information Sciences, Vol. 178, No. 13, pp. 2751–2779. DOI: 10.1016/j.ins.2008.02.012.
37. **Serrano-Guerrero, J., Romero, F. P., Olivas, J. A. (2021).** Fuzzy logic applied to opinion mining: a review. Knowledge-Based Systems, Vol. 222, pp. 107018. DOI: 10.1016/j.knosys.2021.107018.
38. **Daoud, M. S., Shehab, M., Abualigah, L., Thanh, C. (2023).** Hybrid modified chimp optimization algorithm and reinforcement learning for global numeric optimization. Journal of Bionic Engineering, Vol. 20, No. 6, pp. 2896–2915. DOI: 10.1007/s42235-023-00394-2.
39. **Shehab, M., Abualigah, L. (2022).** Opposition-based learning multi-verse optimizer with disruption operator for optimization problems. Soft Computing, Vol. 26, No. 21, pp. 11669–11693. DOI: 10.1007/s00500-022-07470-5.
40. **Tai, K., El-Sayed, A., Biglarbegian, M., Gonzalez, C., Castillo, O., Mahmud, S. (2016).** Review of recent type-2 fuzzy

controller applications. Algorithms, Vol. 9, No. 2, pp. 39. DOI: 10.3390/a9020039.

41. **Ontiveros, E., Melin, P., Castillo, O. (2020).** Comparative study of interval type-2 and general type-2 fuzzy systems in medical diagnosis. Information Sciences, Vol. 525, pp. 37–53. DOI: 10.1016/j.ins.2020.03.059.
42. **Moreno, J. E., Sanchez, M. A., Mendoza, O., Rodríguez-Díaz, A., Castillo, O., Melin, P., Castro, J. R. (2020).** Design of an interval type-2 fuzzy model with justifiable uncertainty. Information Sciences, Vol. 513, pp. 206–221. DOI: 10.1016/j.ins.2019.10.042.
43. **Guerrero, M., Valdez, F., Castillo, O. (2022).** Comparative study between type-1 and interval type-2 fuzzy systems in parameter adaptation for the cuckoo search algorithm. Symmetry, Vol. 14, No. 11, pp. 2289. DOI: 10.3390/sym14112289.
44. **Cuevas, F., Castillo, O., Cortés-Antonio, P. (2022).** Generalized type-2 fuzzy parameter adaptation in the marine predator algorithm for fuzzy controller parameterization in mobile robots. Symmetry, Vol. 14, No. 5, pp. 859. DOI: 10.3390/sym14050859.

Article received on 10/02/2024; accepted on 01/05/2024.
**Corresponding author is Marylu L. Lagunes.*

ISSN 2007-9737

Adjustment of Convolutional and Hidden Layers Using Type-1 Fuzzy Logic Applied to Diabetic Retinopathy Classification

Rodrigo Cordero-Martínez, Daniela Sánchez*, Patricia Melin

Tecnológico Nacional de México, Campus Tijuana,
Mexico

{rodrigo.cordero201, daniela.sanchez, pmelin}@tectijuana.mx

Abstract. Vision problems are common in patients with diabetes mellitus (DM) because they may suffer from diabetic retinopathy (DR). Because the symptoms of this condition are not easy to detect without the intervention of an expert technician, the use of convolutional neural networks (CNN) has been implemented to speed up the process of analyzing retina images. Due to the good results of this technology, efforts have been made to combine it with other technologies. In this paper, we present the use of an intelligent hybrid system that uses CNNs and Fuzzy Logic with the aim of improving the accuracy obtained. The implementation of fuzzy logic to adjust the hyperparameters of the network allowed us to obtain a mean of 0.9526 with a standard deviation of 0.008521158 in the binary case study, while in the multiclass case study we obtained a mean of 0.7299 and a standard deviation of 0.015614013, offering better results when fuzzy logic is combined compared to when not.

Keywords. Convolutional neural network, fuzzy logic, image pre-processing.

1 Introduction

Individuals afflicted with diabetes mellitus (DM), irrespective of whether it manifests as type 1 or type 2, face a significantly elevated likelihood of developing diabetic retinopathy (DR) [1-2]. Those grappling with this condition may experience visual impairments or, in severe cases, complete blindness [3-4]. A prognostic study indicates that within a mere decade, the prevalence of this ailment is anticipated to double compared to patient counts from a decade prior [5].

Consequently, the incidence of vision impairment is poised to witness a twofold increase. Several factors contribute to the expeditious proliferation of DR cases [6-7]. The foremost factor is the onset age of this condition, which has been documented in DM patients as young as 20 years old [8].

Another critical concern in the context of DR is the dearth of expert technicians that are adept at diagnosing the ailment [9-10]. In response to this challenge, diverse computer technologies have been harnessed to facilitate prompt detection and informed medical decision-making for patients.

Researchers have leveraged advanced tools like deep learning, particularly neural networks, to expedite the detection process not only for DR but also for other diseases, concurrently diminishing the margin of error [11-14].

Techniques such as convolutional neural networks (CNNs) and deep learning methodologies have proven highly efficacious in DR detection, with optimal outcomes realized through their application [15-17]. Many researchers have expedited their work by utilizing pre-trained CNNs, yet bespoke architectures tailored to specific issues have exhibited enhanced detection accuracy, incorporating techniques such as optimization algorithms and fuzzy logic [18].

This study commenced by meticulously selecting the foundational CNN model and the most appropriate pre-processing technique for the designated database. Exhaustive research ensured the identification of the pre-processing method that exhibited superior performance in prior studies.

Similarly, the base CNN model was chosen based on the favorable outcomes discerned in the research. The subsequent phase involved the integration of a fuzzy inference system to optimize the layers of the CNN architecture for DR detection. Despite the prevalent use of fuzzy logic

Table 1. Multiple image distribution of APTOS 2019

Distribution Name	Number of images
Retina without DR	1805
No proliferative mild	370
No proliferative moderate	999
No proliferative severe	193
Proliferative	295
Retina with any proliferative stage of DR	1857

in other studies for classifying diverse databases, its application in the detection and classification of DR remains relatively unexplored.

Section 2 of this paper elucidates the related work conducted by various authors. Section 3 provides comprehensive explanations of fundamental concepts, complemented by illustrative examples, to enhance comprehension of this study.

Section 4 delineates and expounds upon the methods employed in this research. Section 5 presents the results gleaned from the conducted experiments, and finally, Section 6 articulates the conclusions derived from the study.

2 Related Work

Increasing the precision of neural networks either for classification or prediction has been a priority for experts in recent years [19-20], due to this many works have been written with different implementations or own methods that seek the same objective.

In the work of [21] we sought to modify the depth of the network, size and number of convolutional filters and number of neurons in the hidden layers. In this work it was concluded that, depending on the data set used, the number of filters used in the convolution layer can be increased or decreased, so increasing the number of filters will not necessarily improve the results.

Do not forget that a convolutional neural network contains the same hidden layers as a traditional neural network. The work of the author [22] shows us the different behaviors that modifying the number of neurons in each of the hidden layers can result in an increase of precision.

To carry out the modifications to the hyperparameters of the CNNs, the use of optimization algorithms has been implemented that allow experts to speed up the process, one of them being the genetic algorithm. Thanks to this technology, prior work can be carried out with the APTOS 2019 [23] database. Using this algorithm, a CNN model was created that improves the accuracy of diabetic retinopathy classification.

3 Basic Concepts

In the previous section, some terms were mentioned that may not be familiar to those who do not work with the use of intelligent hybrid systems. So, for a complete understanding of this work, this section will present the information necessary to understand the work in its entirety.

3.1 Artificial Neural Networks

One of the most widely employed machine learning tools for disease detection is the supervised artificial neural network. This specific neural network variant enables expert technicians to train the network using a labeled database [24].

When utilizing images in the learning phase, it becomes imperative to specify pertinent information for accurate future classification. This underscores the necessity for the supervision of an expert technician in the realm of image topics.

However, there exists a type of neural network capable of performing this task with convolutional filters: convolutional neural networks [25]. The design of a convolutional neural network model closely resembles that of a feed-forward neural network, with the differentiating factor becoming evident after the input stage [26].

3.1.1. Convolutional Layer

It is the initial layer in the CNN architecture, facilitates the recognition of key characteristics within the input images [27].

Consequently, the network eliminates the requirement for an expert technician to apply preprocessing methods to the images. To achieve

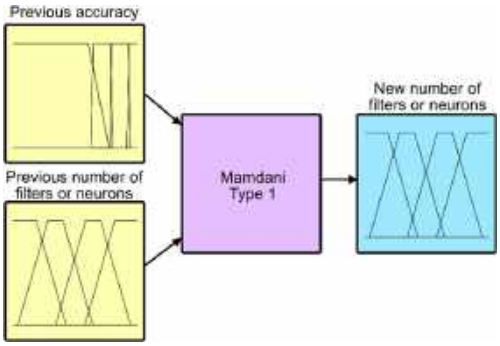


Fig. 1. Graphical representation of FIS

Table 2. Ranges of the new filters number

Convolutional Layer Number	Range
1	[16-32]
2	[32-64]
3	[64-128]
4	[128-256]
5	[256-512]

Table 3. Ranges of the new neurons number

Fully Connected Layer Number	Range
1 (Binary Study Case)	[64 - 128]
2 (Binary Study Case)	[128 - 256]
3 (Binary Study Case)	[256 - 512]
1 (Multiclass Study Case)	[64 - 512]

this, the convolutional layer necessitates a kernel to derive a new matrix.

3.1.2. ReLU Function

The Rectified Linear Unit Function (ReLU) plays a crucial role in Convolutional Neural Networks (CNNs). Activation functions are essential after each individual neuron, and one of the widely utilized functions for CNNs is the ReLU function [28].

ReLU permits the activation of every positive number, thereby reducing the time required for experimentation.

3.1.3. MaxPooling Layer

The activation function serves to expedite the training time, although the image size remains constant, and not all pixels carry equal significance. To address this, a pooling method is employed, with the MaxPooling method being one of the most prevalent in the domain of convolutional neural networks [29].

3.2 Fuzzy Logic

Fuzzy logic is a logic that allows you to reach "reasoned" conclusions based on ambiguous or imprecise information [30]. One of the great contributions of this logic is that it allows us to model situations or behaviors that are vague in themselves, that is, it adapts better to reality than classic logic, where there are only two values to decide [31].

Human reasoning does not react in a classical logic manner, but on the contrary, evaluates the environment and based on the weights of each of the variables decides, so fuzzy logic is more suitable to try to emulate such mental behavior [32].

It has been used for the development of a countless number of applications of all kinds such as medicine and bioinformatics [33]. The primary emphasis of this study lies in hybrid systems, signifying the necessity of incorporating two or more distinct techniques to formulate the proposed method.

The utilization of both CNN and fuzzy logic has been a recurrent theme in previous works, with a noticeable increase in its prevalence over the years [34]. The amalgamation of convolutional neural networks and fuzzy logic typically involves the incorporation of optimization algorithms, such as genetic algorithms or particle swarm optimization, aimed at optimizing the parameters associated with each technology [35].

4 Proposed Methods

In this section, we will consider each of the concepts explained in the previous section. First, the architectures of the CNN models from a previous work to which the proposed method will

Table 4. Fuzzy rules

Fuzzy Rule Number	Fuzzy Rule
1	If (accuracy is very_bad) and (old_filters is very_few) then (new_filters is a_lot)
2	If (accuracy is bad) and (old_filters is very_few) then (new_filters is many)
3	If (accuracy is good) and (old_filters is very_few) then (new_filters is few)
4	If (accuracy is excellent) and (old_filters is very_few) then (new_filters is very_few)
5	If (accuracy is very_bad) and (old_filters is few) then (new_filters is a_lot)
6	If (accuracy is bad) and (old_filters is few) then (new_filters is many)
7	If (accuracy is good) and (old_filters is few) then (new_filters is very_few)
8	If (accuracy is excellent) and (old_filters is few) then (new_filters is few)
9	If (accuracy is very_bad) and (old_filters is many) then (new_filters is very_few)
10	If (accuracy is bad) and (old_filters is many) then (new_filters is few)
11	If (accuracy is good) and (old_filters is many) then (new_filters is a_lot)
12	If (accuracy is excellent) and (old_filters is many) then (new_filters is many)
13	If (accuracy is very_bad) and (old_filters is a_lot) then (new_filters is very_few)
14	If (accuracy is bad) and (old_filters is a_lot) then (new_filters is few)
15	If (accuracy is good) and (old_filters is a_lot) then (new_filters is many)
16	If (accuracy is excellent) and (old_filters is a_lot) then (new_filters is a_lot)

be applied will be detailed. Afterwards, the data from the APTOS 2019 database, characteristics and the two distributions necessary for the case studies of this work will be presented.

Then the explanation of the preprocessing applied to the database will be gone into detail. And finally, the creation and implementation of a fuzzy inference system that will allow us to modify the number of filters and neurons in the networks.

4.1 Neural Network Models

There are two CNN models obtained on previous work [36]. The first model, focused on binary study case, has an input size of 256x256x3 (width, height, and depth) and 5 convolutional layers, some on them have MaxPooling layer after it, but not all. Also, the model has 3 fully connected layers with different number of neurons.

Finally, the model has a Sigmoid activation function because it is for a binary study case. The second model, focused on multiclass study case, has the same input size and 5 convolutional layers with different values on its hyperparameters (MaxPooling size dropout if applies). Also, the model has 1 fully connected layer. Finally, the model has a Softmax activation function because it is for multiclass study case.

4.2 APTOS 2019 Database

This DR database has 3662 labeled images can be used for training and validation [37]. The database has 5 different classes that represent the damage caused by the disease that are used for multiclass study case [38], but 4 classes can be combined to obtain just 2 total classes for binary study case: healthy retina images and retina with diabetic retinopathy images [39].

The database has images with different kind of noise and no image has the same size. In Table 1, the distribution of the images for this work can be observed.

4.3 Preprocessing Method

This approach involves removing interfering pixels from the background and completely isolating the retina in the image. To achieve this goal, it is essential to transform color images to grayscale.

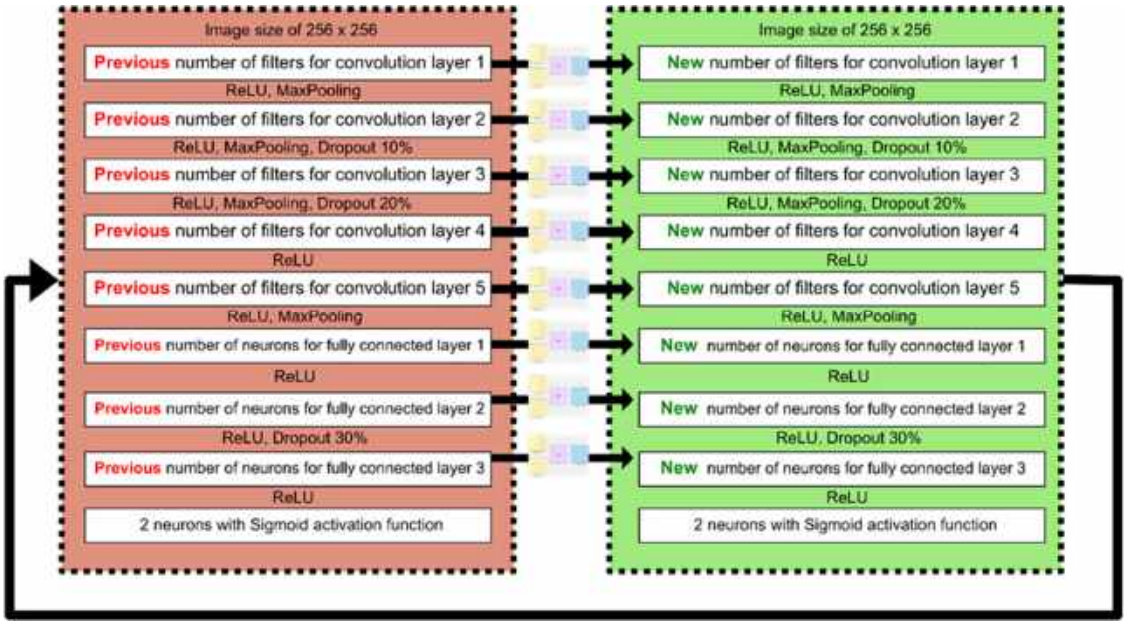


Fig. 2. Graphical representation of the proposed method

By using a grayscale representation, conversion to a binary image becomes feasible. The process of converting the grayscale image to binary involves the selection of each pixel to refine the image.

The amount of luminosity present in the pixel must exceed a defined threshold to avoid considering it as noise and instead take advantage of it for precise retinal extraction. With the resulting binary image, the next step consists of locating and identifying the retina.

Thanks to the binary image, this task is simplified since it only involves identifying the most prominent shape. Once the retina has been detected, its position is extracted and used to isolate the retina from the original images.

Finally, black pixels are inserted as necessary to achieve an image with uniform dimensions in width and height. This method has yielded favorable results compared to other preprocessing techniques [40].

4.4 Fuzzy Inference System Description

To start, both APTOS 2019 study cases require a Mamdani Type-1 fuzzy inference system (FIS). All membership functions are trapezoidal functions.

The FIS comprises two inputs: the accuracy achieved with the current quantity of filters or neurons, and the second input pertains to the current quantity of filters or neurons.

The FIS yields a single output, which signifies the number of filters for the convolutional layer or neurons for the fully connected layer. Each input and output have 4 membership functions. The graphical representation of the FIS is depicted in Fig. 1. The accuracy value is normalized within the range of 0 to 1.

The quantity of filters or neurons is contingent upon the convolutional layer number or fully connected layer, with specified ranges detailed in Table 2 and Table 3 respectively.

The chosen Defuzzification Method is Centroid. In the context of these experiments, the FIS incorporates 16 fuzzy if-then rules obtained by trial and error, which are outlined in Table 4. A graphical representation of the proposed method can be observed on Fig. 2.

Equations of the FIS can be observed on Eq. 1-8 where Eq. 1-4 are for the input of accuracy and Eq. 5-8 are for the input and output of the number of filters or neurons.

$$\mu_{\text{very\_bad}}(x) = \begin{cases} 0, & x \le -0.2 \\ \frac{x-(-0.2)}{-0.1-(-0.2)}, & -0.2 \le x \le -0.1 \\ 1, & -0.1 \le x \le 0.7 \\ \frac{0.8-x}{0.8-0.7}, & 0.7 \le x \le 0.8 \\ 0, & 0.8 \le x \end{cases}, \quad (1)$$

$$\mu_{\text{bad}}(x) = \begin{cases} 0, & x \le 0.65 \\ \frac{x-0.65}{0.72-0.65}, & 0.65 \le x \le 0.72 \\ 1, & 0.72 \le x \le 0.83 \\ \frac{0.9-x}{0.9-0.83}, & 0.83 \le x \le 0.9 \\ 0, & 0.9 \le x \end{cases}, \quad (2)$$

$$\mu_{\text{good}}(x) = \begin{cases} 0, & x \le 0.8 \\ \frac{x-0.8}{0.85-0.8}, & 0.8 \le x \le 0.85 \\ 1, & 0.85 \le x \le 0.95 \\ \frac{1-x}{1-0.95}, & 0.9 \le x \le 1 \\ 0, & 1 \le x \end{cases}, \quad (3)$$

$$\mu_{\text{excellent}}(x) = \begin{cases} 0, & x \le 0.9 \\ \frac{x-0.9}{0.96-0.9}, & 0.9 \le x \le 0.96 \\ 1, & 0.96 \le x \le 1.042 \\ \frac{1.375-x}{1.375-1.042}, & 1.042 \le x \le 1.375 \\ 0, & 1.375 \le x \end{cases}, \quad (4)$$

$$\mu_{\text{very\_few}}(x) = \begin{cases} 0, & x \le -0.375 \\ \frac{x-(-0.375)}{-0.04-(-0.375)}, & -0.375 \le x \le -0.04 \\ 1, & -0.04 \le x \le 0.1 \\ \frac{0.4-x}{0.4-0.1}, & 0.1 \le x \le 0.4 \\ 0, & 0.4 \le x \end{cases} \quad (5)$$

$$\mu_{\text{few}}(x) = \begin{cases} 0, & x \le -0.005 \\ \frac{x-(-0.005)}{0.3-(-0.005)}, & -0.005 \le x \le 0.3 \\ 1, & 0.3 \le x \le 0.4 \\ \frac{0.7-x}{0.7-0.4}, & 0.4 \le x \le 0.7 \\ 0, & 0.7 \le x \end{cases} \quad (6)$$

$$\mu_{\text{many}}(x) = \begin{cases} 0, & x \le 0.3 \\ \frac{x-0.3}{0.6-0.3}, & 0.3 \le x \le 0.6 \\ 1, & 0.6 \le x \le 0.7 \\ \frac{1-x}{1-0.7}, & 0.7 \le x \le 1 \\ 0, & 1 \le x \end{cases} \quad (7)$$

$$\mu_{\text{a\_lot}}(x) = \begin{cases} 0, & x \le 0.6 \\ \frac{x-0.6}{0.9-0.6}, & 0.6 \le x \le 0.9 \\ 1, & 0.9 \le x \le 1.015 \\ \frac{1.315-x}{1.315-1.015}, & 1.015 \le x \le 1.315 \\ 0, & 1.315 \le x \end{cases} \quad (8)$$

5 Experimental Results

In this section, we are going to bring together each of the concepts and methods proposed for distributed experimentation in two study cases: binary and multiclass.

5.1 Experiments for APTOS 2019 Binary Study Case

Two experiments were conducted; the initial one involved employing the CNN model derived through the hierarchical genetic algorithm as documented in prior research [41]. The second experiment utilized the CNN model acquired through the previously explained FIS.

Each experiment was iterated 30 times, maintaining consistent hyperparameters: 10 epochs, utilization of the APTOS 2019 database, and the Adam optimizer algorithm.

In the first experiment, the mean accuracy recorded was 0.9021, accompanied by a standard deviation of 0.108434797. Conversely, for the second experiment, the mean accuracy achieved was 0.9526, with a standard deviation of 0.008521158. Detailed results for each iteration are presented in Table 5.

5.1.1. Box Plot for Binary Study Case

One box plot was made to observe the comparison of the values. Box plot for the binary study case can be observed on Fig. 3.

5.1.2. Hypothesis Testing for Binary Study Case

Based on the results observed in Table 5, the hypothesis testing will be between mean accuracy and standard deviation obtained by the first experiment and the second one. The experiment of this present work got a higher mean accuracy, so, our statement is that the experiment with the CNN

Table 5. Results for binary study case experimentation

Experiment Number	Genetic Algorithm Accuracy	Fuzzy Logic Accuracy	Experiment Number	Genetic Algorithm Accuracy	Fuzzy Logic Accuracy
1	0.938608468	0.934515715	16	0.927694380	0.960436583
2	0.929058671	0.956343770	17	0.889495254	0.938608468
3	0.911323309	0.960436583	18	0.931787193	0.954979539
4	0.507503390	0.952251017	19	0.950886786	0.949522495
5	0.920873106	0.956343770	20	0.960436583	0.956343770
6	0.930422902	0.961800814	21	0.953615308	0.952251017
7	0.939972699	0.961800814	22	0.897680759	0.972714841
8	0.937244177	0.942701221	23	0.931787193	0.949522495
9	0.924965918	0.957708061	24	0.904502034	0.954979539
10	0.938608468	0.956343770	25	0.934515715	0.935879946
11	0.937244177	0.960436583	26	0.916780353	0.950886786
12	0.937244177	0.960436583	27	0.929058671	0.946793973
13	0.939972699	0.952251017	28	0.934515715	0.945429742
14	0.946793973	0.952251017	29	0.507503390	0.954979539
15	0.930422902	0.939972699	30	0.933151424	0.948158264

Table 6. Results for multiclass study case experimentation

Experiment Number	Genetic Algorithm Accuracy	Fuzzy Logic Accuracy	Experiment Number	Genetic Algorithm Accuracy	Fuzzy Logic Accuracy
1	0.728512943	0.717598915	16	0.688949525	0.740791261
2	0.706684828	0.738062739	17	0.703956366	0.720327437
3	0.714870393	0.733969986	18	0.731241465	0.736698508
4	0.712141871	0.712141871	19	0.710777640	0.744884014
5	0.710777640	0.731241465	20	0.703956366	0.727148712
6	0.714870393	0.733969986	21	0.735334218	0.713506162
7	0.727148712	0.723055959	22	0.728512943	0.720327437
8	0.736698508	0.743519783	23	0.739427030	0.731241465
9	0.740791261	0.739427030	24	0.739427030	0.729877234
10	0.750341058	0.729877234	25	0.718963146	0.725784421
11	0.724420190	0.742155552	26	0.690313756	0.727148712
12	0.725784421	0.710777640	27	0.712141871	0.727148712
13	0.727148712	0.724420190	28	0.706684828	0.721691668
14	0.688949525	0.725784421	29	0.720327437	0.735334218
15	0.716234624	0.736698508	30	0.717598915	0.753069580

model obtained by the fuzzy system inference offers a bigger mean accuracy than the offered by the experiment with the CNN model obtained by the hierarchical genetic algorithm for binary study case. Using an Alpha value of 0.05, the critical value obtained must be more than 1.96 to reject the null hypothesis. The score of the statistic test is 2.5240, meaning that the null hypothesis is rejected and there is enough evidence to support the claim.

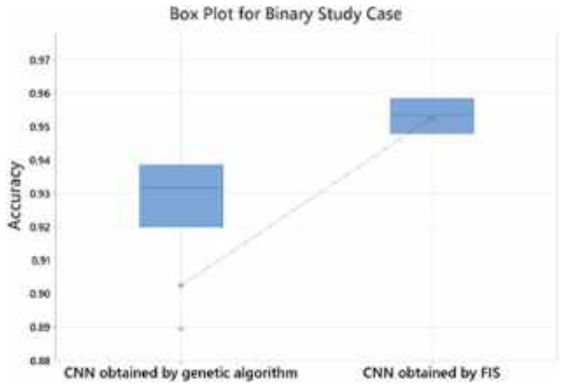


Fig. 3. Box plot for binary study case

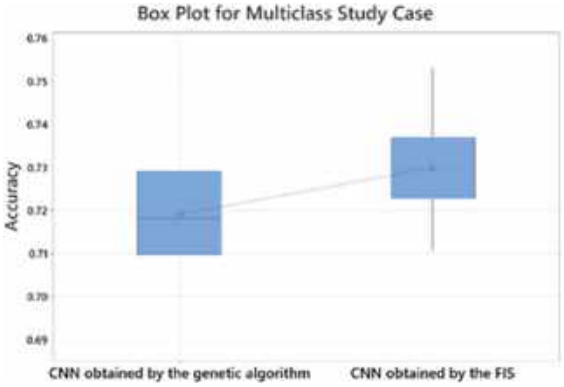


Fig. 4. Box plot for multiclass study case

5.2 Experiments for APTOS 2019 Multiclass Study

In the same way as previous experimentation, two experiments were conducted; the first one involved employing the CNN model obtained through the hierarchical genetic algorithm [42] and the second experiment utilized the CNN model acquired through the FIS. Each experiment was iterated 30 times, maintaining consistent hyperparameters: 10 epochs, utilization of the APTOS 2019 database, and the Adam optimizer algorithm.

In the first experiment, the mean accuracy recorded was 0.7191, accompanied by a standard deviation of 0.010199619. Conversely, for the second experiment, the mean accuracy achieved was 0.7299, with a standard deviation of 0.015614013. Detailed results for each iteration are presented in Table 6.

5.2.1. Box Plot for Multiclass Study Case

In the same way as the previous experiment, one box plot was made to observe the comparison of the values. Box plot for the multiclass study case can be observed on Fig. 4.

5.1.2. Hypothesis Testing for Multiclass Study Case

Based on the results observed in Table 6, the hypothesis testing will be between mean accuracy and standard deviation obtained by the first experiment and the second one.

The experiment of this present work got a higher mean accuracy, so, our statement is that the experiment with the CNN model obtained by the fuzzy system inference offers a bigger mean accuracy than the offered by the experiment with the CNN model obtained by the hierarchical genetic algorithm for multiclass study case.

Using an Alpha value of 0.05, the critical value obtained must be more than 1.96 to reject the null hypothesis. The score of the statistic test is 3.1786, meaning that the null hypothesis is rejected and there is enough evidence to support the claim.

6 Conclusions

In this study, the focus was on employing a Mamdani Type 1 fuzzy inference system to determine the filter numbers based on the previous filter values and the obtained accuracy. Before implementing the proposed method, the mean accuracy and standard deviation of the base CNN model were calculated for comparative analysis. Subsequently, the proposed method was integrated into a pre-existing CNN model.

After the generation of the new CNN model, the FIS was used iteratively to refine the CNN model, obtaining the best CNN model, mean precision, and standard deviation. There is room for improvement in the construction of the FIS such as the number of variables, rules and membership functions, so as future work the current work can be taken and implemented the respective improvements and seek a higher average precision with a reduced standard deviation.

Finally, APTOS 2019 serves as a valuable database in addressing real-world problems, yet it is not the exclusive dataset where the proposed

method could find application. As future work, we plan to consider different metaheuristics for optimizing the method, as in [43-48]. Also, elevate the use of fuzzy logic to type-2, like it is done in several recent works [49-54].

Acknowledgments

We would like to express our gratitude to the CONAHCYT and Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

1. **Alex, S. A., Nayahi, J. J. V., Shine, H., Gopirekha, V. (2021).** Deep convolutional neural network for diabetes mellitus prediction. Neural Computing and Applications, Vol. 34, No. 2, pp. 1319–1327. DOI: 10.1007/s00521-021-06431-7.
2. **Milluzzo, A., Maugeri, A., Barchitta, M., Sciacca, L., Agodi, A. (2021).** Epigenetic mechanisms in type 2 diabetes retinopathy: a systematic review. International Journal of Molecular Sciences, Vol. 22, No. 19, pp.10502. DOI: 10.3390/ijms221910502.
3. **Singh, S., Rathore, B., Das, H., Agrawal, S., Bhutia, D., Maan, V., Jangir, S. K., Kumar, M. (2022).** An improved convolutional neural network for classification of type-2 diabetes mellitus. Algorithms for Intelligent Systems, pp.417–424. DOI: 10.1007/978-981-19-2065- 3_45.
4. **Flaxel, C. J., Adelman, R. A., Bailey, S. T., Fawzi, A., Lim, J. I., Vemulakonda, G. A., Ying, G. (2020).** Diabetic retinopathy preferred practice pattern®. Ophthalmology, Vol. 127, No. 1, pp. 66–145. DOI: 10.1016/j.ophtha. 2019.09.025.
5. **Rocha, D. A. D., Ferreira, F. M. F., Peixoto, Z. M. A. (2022).** Diabetic retinopathy classification using VGG16 neural network. Research on Biomedical Engineering, Vol. 38, No. 2, pp. 761–772. DOI: 10.1007/s42600-022-00200-8.
6. **Correia, J. L., Moreira-Bessa, W. (2022).** Intelligent control with artificial neural networks for automated insulin delivery systems. Bioengineering, Vol. 9, No. 11, pp. 664. DOI: 10.3390/bioengineering9110664.
7. **Ranjan, R., Gupta, S., Sharma, S., Alatba, S. R., Sreeram, A., Fernandez, S. (2023).** Artificial neural network based categorization of diabetes mellitus and its comparison with k-nearest neighbors algorithm. Proceedings of the 3rd International Conference on Advance Computing and Innovative Technologies in Engineering, pp. 1099–1101 DOI: 10.1109/ icacite57410.2023. 10182961.
8. **Prabhu, P., Selvabharathi, S. (2019).** Deep belief neural network model for prediction of diabetes mellitus. 3rd International Conference on Imaging, Signal Processing and Communication, pp. 138–142 DOI: 10.1109/ icispc.2019.8935838.
9. **Fung, T. H., Patel, B., Wilmot, E. G., Amoaku, W. M. (2022).** Diabetic retinopathy for the non-ophthalmologist. Clinical Medicine, Vol. 22, No. 2, pp. 112–116. DOI: 10.78 61/ clinmed.2021-0792.
10. **Hatmal, M. M., Abderrahman, S. M., Nimer, W., Al-Eisawi, Z., Al-Ameer, H. J., Al-Hatamleh, M. A. I., Mohamud, R., Alshaer, W. (2020).** Artificial neural networks model for predicting type 2 diabetes mellitus based on VDR gene FoKI polymorphism, lipid profile and demographic data. Biology, Vol. 9, No. 8, pp. 222. DOI: 10.3390/biology9080222.
11. **Hosseinzadeh, M., Ahmed, O. H., Ghafour, M. Y., Safara, F., Hama, H. K., Ali, S., Vo, B., Chiang, H. (2020).** A multiple multilayer perceptron neural network with an adaptive learning algorithm for thyroid disease diagnosis in the internet of medical things. The Journal of Supercomputing, Vol. 77, No. 4, pp. 3616–3637. DOI: 10.1007/s11227-020-03404- w.
12. **Janghel, R., Rathore, Y. (2021).** Deep convolution neural network based system for early diagnosis of alzheimer's disease. IRBM, Vol. 42, No. 4, pp. 258–267. DOI: 10.1016/ j.irbm.2020.06.006.
13. **Irmak, E. (2020).** A novel deep convolutional neural network model for covid-19 disease

detection. Medical Technologies Congress, pp. 1–4 DOI: 10.1109/tiptekno50054. 2020.9299286.

14. **Alharbi, A., Alghahtani, M. (2018).** Using genetic algorithm and ELM neural networks for feature extraction and classification of type 2-diabetes mellitus. Applied Artificial Intelligence, Vol. 33, No. 4, pp. 311–328. DOI: 10.1080/08839514.2018.1560545.
15. **Kandel, I., Castelli, M. (2020).** Transfer learning with convolutional neural networks for diabetic retinopathy image classification. a review. Applied Sciences, Vol. 10, No. 6, pp. 2021. DOI: 10.3390/app10062021.
16. **Saeed, F., Hussain, M., Aboalsamh, H. A. (2021).** Automatic diabetic retinopathy diagnosis using adaptive fine-tuned convolutional neural network. IEEE Access, Vol. 9, pp. 41344–41359. DOI: 10.1109/acc ess.2021.3065273.
17. **Wen, D., Li, P., Zhou, Y., Sun, Y., Xu, J., Liu, Y., Li, X., Li, J., Bian, Z., Wang, L. (2020).** Feature classification method of resting-state EEG signals from amnestic mild cognitive impairment with type 2 diabetes mellitus based on multi-view convolutional neural network. IEEE Transactions on Neural Systems and Rehabilitation Engineering, Vol. 28, No. 8, pp. 1702–1709. DOI: 10.1109/tnsre .2020. 3004462.
18. **Abdolrasol, M. G. M., Hussain, S. M. S., Ustun, T. S., Sarker, M. R., Hannan, M. A., Mohamed, R., Ali, J. A., Mekhilef, S., Milad, A. (2021).** Artificial neural networks based optimization techniques: a review. Electronics, Vol. 10, No. 21, pp. 2689. DOI: 10.3390/electro nics10212689.
19. **Baioletti, M., Bari, G. D., Milani, A., Poggioni, V. (2020).** Differential evolution for neural networks optimization. Mathematics, Vol. 8, No. 1, pp. 69. DOI: 10.3390/math8 010069.
20. **Sinha, T., Verma, B., Haidar, A. (2017).** Optimization of convolutional neural network parameters for image classification. IEEE Symposium Series on Computational Intelligence, pp. 1–7 DOI: 10.1109/ssci.2017. 8285338.
21. **Uzair, M., Jamil, N. (2020).** Effects of hidden layers on the efficiency of neural networks. IEEE 23rd International Multitopic Conference, pp. 1–6 DOI: 10.1109/inmic50486.2020. 9318195.
22. **Melin, P., Sánchez, D., Cordero-Martínez, R. (2023).** Particle swarm optimization of convolutional neural networks for diabetic retinopathy classification. Studies in Computational Intelligence, pp. 237–252. DOI: 10.1007/978-3-031-22042-5_14.
23. **Singh, P., Singh, P., Farooq, U., Khurana, S. S., Verma, J. K., Kumar, M. (2023).** Cottonleafnet: cotton plant leaf disease detection using deep neural networks. Multimedia Tools and Applications, Vol. 82, No. 24, pp. 37151–37176. DOI: 10.1007/s110 42-023-14954-5.
24. **Bala, D., Hossain, M. S., Hossain, M. A., Abdullah, M. I., Rahman, M. M., Manavalan, B., Gu, N., Islam, M. S., Huang, Z. (2023).** Monkeynet: A robust deep convolutional neural network for monkeypox disease detection and classification. Neural Networks, Vol. 161, pp. 757–775. DOI: 10.1016/j.neunet. 2023.02.022.
25. **Gadekallu, T. R., Khare, N., Bhattacharya, S., Singh, S., Maddikunta, P. K. R., Srivastava, G. (2020).** Deep neural networks to predict diabetic retinopathy. Journal of Ambient Intelligence and Humanized Computing, Vol. 14, No. 5, pp. 5407–5420. DOI: 10.1007/s12652-020-01963-7.
26. **Jiwani, N., Gupta, K., Afreen, N. (2022).** A convolutional neural network approach for diabetic retinopathy classification. IEEE 11th International Conference on Communication Systems and Network Technologies, pp. 357–361. DOI: 10.1109/csnt54456. 2022.9787577.
27. **Chen, C., Min, F., Zhang, Y., Bao, H. (2023).** Relu-type Hopfield neural network with analog hardware implementation. Chaos, Solitons and Fractals, Vol. 167, pp. 113068. DOI: 10.10 16/j.chaos.2022.113068.
28. **Zhang, J., Zhang, Z., Wang, L., Zhou, L., Zhang, X., Liu, M., Wu, W. (2023).** Kernel-based feature aggregation framework in point cloud networks. Pattern Recognition, Vol. 139,

pp. 109439. DOI: 10.1016/j.patcog.2023. 109439.

29. **Kacimi, M. A., Guenounou, O., Brikh, L., Yahiaoui, F., Hadid, N. (2020).** New mixed-coding PSO algorithm for a self-adaptive and automatic learning of Mamdani fuzzy rules. Engineering Applications of Artificial Intelligence, Vol. 89, pp. 103417. DOI: 10.10 16/j.engappai.2019.103417.
30. **Kaur, J., Khehra, B. S. (2021).** Fuzzy logic and hybrid based approaches for the risk of heart disease detection: State-of-the-art review. Journal of The Institution of Engineers (India): Series B, Vol. 103, No. 2, pp. 681–697. DOI: 10.1007/s40031-021-00644-z.
31. **Ouifak, H., Idri, A. (2023).** On the performance and interpretability of Mamdani and Takagi-Sugeno-Kang based neuro-fuzzy systems for medical diagnosis. Scientific African, Vol. 20, pp. e01610. DOI: 10.1016/j.s ciaf.2023.e01610.
32. **Kamel, A. A., Zahran, F. (2020).** A fuzzy decision support system for diagnosis of some liver diseases in educational medical institutions. International Journal of Fuzzy Logic and Intelligent Systems, Vol. 20, No. 4, pp. 358–368. DOI: 10.5391/ij fis.2020. 20.4.358.
33. **Deng, Y., Ren, Z., Kong, Y., Bao, F., Dai, Q. (2017).** A hierarchical fused fuzzy deep neural network for data classification. IEEE Transactions on Fuzzy Systems, Vol. 25, No. 4, pp. 1006–1012. DOI: 10.1109/tfuzz .2016. 2574915.
34. **Melin, P., Sánchez, D., Monica, J. C., Castillo, O. (2021).** Optimization using the firefly algorithm of ensemble neural networks with type-2 fuzzy integration for COVID-19 time series prediction. Soft Computing, Vol. 27, No. 6, pp. 3245–3282. DOI: 10.1007/ s00500-020-05549-5.
35. **Cordero-Martínez, R., Sánchez, D., Melin, P. (2023).** Optimizing a convolutional neural network with a hierarchical genetic algorithm for diabetic retinopathy detection. Studies in Computational Intelligence, pp. 199–208. DOI: 10.1007/978-3-031-22042-5_11.
36. **Chetoui, M., Akhloufi, M. A. (2020).** Explainable diabetic retinopathy using efficientnet. 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 1966–1969. DOI: 10.1109/embc44109.2020.9175664.
37. **Bodapati, J. D., Shaik, N. S., Naralasetti, V. (2021).** Composite deep neural network with gated-attention mechanism for diabetic retinopathy severity classification. Journal of Ambient Intelligence and Humanized Computing, Vol. 12, No. 10, pp. 9825–9839. DOI: 10.1007/s12652-020-02727-z.
38. **Macsik, P., Pavlovicova, J., Goga, J., Kajan, S. (2022).** Local binary CNN for diabetic retinopathy classification on fundus images. Acta Polytechnica Hungarica, Vol. 19, No. 7, pp. 27–45. DOI: 10.12700/aph.19.7.2022.7.2.
39. **Cordero-Martínez, R., Sánchez, D., Melin, P. (2022).** Comparison of image pre-processing for classifying diabetic retinopathy using convolutional neural networks. Lecture Notes in Networks and Systems, pp. 194–204. DOI: 10.1007/978-3-030-96305-7_18.
40. **Cordero-Martínez, R., Sánchez, D., Melin, P. (2022).** Hierarchical genetic optimization of convolutional neural models for diabetic retinopathy classification. International Journal of Hybrid Intelligent Systems, Vol. 18, No. 1-2, pp. 97–109. DOI: 10.3233/his-220004.
41. **Amador-Angulo, L., Ochoa, P., Peraza, C., Castillo, O. (2023).** Fuzzy dynamic adaptation of an artificial fish swarm algorithm for the optimization of benchmark functions. Studies in Computational Intelligence, pp. 99–114. DOI: 10.1007/978-3-031-28999-6_6.
42. **Castillo, O., Lizárraga, E., Soria, J., Melin, P., Valdez, F. (2015).** New approach using ant colony optimization with ant set partition for fuzzy control design applied to the ball and beam system. Information Sciences, Vol. 294, pp. 203–215. DOI: 10.1016/j.ins.2014.09.040.
43. **Amador-Angulo, L., Mendoza, O., Castro, J., Rodríguez-Díaz, A., Melin, P., Castillo, O. (2016).** Fuzzy sets in dynamic adaptation of parameters of a bee colony optimization for controlling the trajectory of an autonomous mobile robot. Sensors, Vol. 16, No. 9, pp. 1458. DOI: 10.3390/s16091458.

44. **Valdez, F., Vazquez, J. C., Melin, P., Castillo, O. (2017).** Comparative study of the use of fuzzy logic in improving particle swarm optimization variants for mathematical functions using co-evolution. Applied Soft Computing, Vol. 52, pp. 1070–1083. DOI: 10.1016/j.asoc.2016.09.024.
45. **Sánchez, D., Melin, P., Castillo, O. (2017).** A grey wolf optimizer for modular granular neural networks for human recognition. Computational Intelligence and Neuroscience, Vol. 2017, pp. 1–26. DOI: 10.1155/2017/4180510.
46. **González, B., Valdez, F., Melin, P., Prado-Arechiga, G. (2015).** Fuzzy logic in the gravitational search algorithm for the optimization of modular neural networks in pattern recognition. Expert Systems with Applications, Vol. 42, No. 14, pp. 5839–5847. DOI: 10.1016/j.eswa.2015.03.034.
47. **Tai, K., El-Sayed, A., Biglarbegian, M., Gonzalez, C., Castillo, O., Mahmud, S. (2016).** Review of recent type-2 fuzzy controller applications. Algorithms, Vol. 9, No. 2, pp. 39. DOI: 10.3390/a9020039.
48. **Ontiveros, E., Melin, P., Castillo, O. (2020).** Comparative study of interval type-2 and general type-2 fuzzy systems in medical diagnosis. Information Sciences, Vol. 525, pp. 37–53. DOI: 10.1016/j.ins.2020.03.059.
49. **Melin, P., Castillo, O. (2003).** A new method for adaptive model-based control of non-linear plants using type-2 fuzzy logic and neural networks. The 12th IEEE International Conference on Fuzzy Systems, Vol. 1, pp. 420–425. DOI: 10.1109/FUZZ.2003.1209400.
50. **Moreno, J. E., Sanchez, M. A., Mendoza, O., Rodríguez-Díaz, A., Castillo, O., Melin, P., Castro, J. R. (2020).** Design of an interval type-2 fuzzy model with justifiable uncertainty. Information Sciences, Vol. 513, pp. 206–221. DOI: 10.1016/j.ins.2019.10.042.
51. **Guerrero, M., Valdez, F., Castillo, O. (2022).** Comparative study between type-1 and interval type-2 fuzzy systems in parameter adaptation for the cuckoo search algorithm. Symmetry, Vol. 14, No. 11, pp. 2289. DOI: 10.3390/sym14112289.
52. **Cuevas, F., Castillo, O., Cortés-Antonio, P. (2022).** Generalized type-2 fuzzy parameter adaptation in the marine predator algorithm for fuzzy controller parameterization in mobile robots. Symmetry, Vol. 14, No. 5, pp. 859. DOI: 10.3390/sym14050859.

Article received on 30/012024; accepted on 23/04/2024.
Corresponding author is Daniela Sánchez.

ISSN 2007-9737

Water Stress Challenges: Mathematical Modeling of Water Resource Management

Valentin Calzada-Ledesma, José Alejandro Cornejo-Acosta*, Blanca Verónica Zúñiga-Núñez

Tecnológico Nacional de México,
Instituto Tecnológico Superior de Purísima del Rincón,
Mexico

alexcornejo@inaoep.mx, {valentin.cl, blanca.zn}@purisima.tecnm.mx

Abstract. Water resource management is an important issue that involves several factors such as economics, social, politician, among others, for its adequate administration. Water can be classified according to its usage purposes since it is used for human consumption, industrial usage, agriculture, etc. Thus, correct strategies to manage this vital liquid are essential to effectively use it. This paper studies water management from a mathematical optimization approach by considering factors and constraints that may suit real-world conditions. The proposed mathematical model is based on the classical transportation problem, which is well-known in the literature. We perform an empirical evaluation of the proposed model using off-the-shelf optimization software over a set of proposed instances, and the results show the feasibility of the proposal. Finally, we discuss the faced challenges in the research and possible future research directions that may help the management of water resources from a computational approach.

Keywords. Water stress, water management optimization, transportation problem, mathematical optimization.

1 Introduction

Water, a vital element for the sustenance of living organisms, faces relentless exploitation due to the current demands and circumstances of humanity. The imperative needs of various economic sectors drive the excessive use of this indispensable resource.

According to [18], global water consumption has been on the rise at approximately 1% annually over the past four decades, and this trend is anticipated to continue until 2050. Despite the inherent renewability of water, its consumption extends beyond human needs, encompassing commercial, industrial, agricultural, livestock, and energy production activities, causing the depletion of water sources, surpassing natural renewal by the hydrological cycle and causing water stress [3].

Efficient water resource management faces numerous limitations and uncertainties, making decision-making very challenging. For example, water is extracted from different sources such as basins, rivers, lakes, etc., and its processing varies depending on the sector in which it is going to be used, so in the end, you have different types of water. That is, the water used for human consumption has different characteristics than that used for irrigation or industry.

Furthermore, the volume of water to be used must be adapted based on the different demands linked to each specific sector (i.e., agricultural, industrial, etc.), population density, geographical conditions, and climate change [16, 26], not to mention that all these aspects require a lot of bureaucracy, which further complicates the management of water resources. The study of the problem of water management using computational tools has been going on for years, for example, the estimation of hydrogeological parameters [7, 9] to establish environmental policies, studies on saltwater intrusion into coastal aquifers by using evolutionary algorithms [1, 2], pollution management in hydrographic basins [24],

and optimal water allocation for crops and irrigation [15], are topics of interest in the computational world.

In addition, alternative methodologies involving the modeling of water management issues using a spectrum of tools have arisen. For instance, in [10] was developed a dynamic model leveraging the expertise of various domain experts, facilitating the selection of optimal water management activities to mitigate water shortages.

In [19] was introduced a hydrological and system dynamics model specifically designed to analyze five distinct scenarios about industrial, agricultural, and domestic water use. [27] contributed to the field by enhancing the Water Resources Ecological Footprint, with a particular emphasis on regional distinctions.

In a different vein, in [29] was introduced a stochastic multi-criteria decision-making framework for Water Resource Management, explicitly considering the challenges posed by uncertainty. In [28] was presented a synthesis of key concepts and categories related to urban drought, elucidating strategies to enhance public awareness, promote flexibility, optimize water management efficiency, ensure reliable and integrated urban water supply, invest in scientific research and strengthen international cooperation.

In [23], was addressed Water Resource Management, specifically targeting irrigation systems through the application of algorithms to calculate limits. In the domain of agricultural water management, in [20] was proposed a generalized spatial fuzzy strategic planning approach, incorporating multi-criteria decision-making.

A strong trend is the optimal design of water distribution systems, whether to improve distribution strategies, pipe rehabilitation, water quality, avoid leaks, optimize the operation of pumps, and also the occurrence of water contamination [13]. Also, parallel evolutionary algorithms have been proposed for similar approaches to optimizing the network design for water distribution.

However, efforts have focused on finding the best system design (at a local level) that maximizes the robustness of the network and at the same time is cost-effective, but the problem of water management is not addressed in a broad context, for example, the challenge of balancing water consumption to promote the replenishment of water resource sources.

Furthermore, if the restrictions mentioned in the previous paragraph (i.e. different supply and demand sources, different types of water, and other restrictions) are added, the problem becomes more difficult to solve. Motivated by the lack of such studies in the literature, we address the challenge of Water Resource Management, where multiple types of source water resources are involved. Besides, other constraints that may suit real-life conditions are considered.

To propose a computational solution, it is necessary to mathematically model the problem. In the state of the art, different approaches could be used to model the problem. However, given that the problem in its simplest form consists of taking water from supply points to demand points, we consider that an approach based on the transportation problem could be a good choice. The Transportation Problem is traditionally linked to the operations research literature [6], which can be seen as the simplification of the objective of minimizing the costs of the carrier that moves certain cargo from one or more origins to their corresponding destinations to satisfy demand.

In this work, we propose a mathematical model as an extension of the transportation problem, where a bipartite graph is established that considers supply nodes, demand nodes, and an associated cost of water transportation. The aim is to minimize the cost of transportation, but our approach does not end there, since as mentioned above, different factors complicate the efficient management of water resources; these factors must be considered in our model to propose solutions that are more in line with reality. It is for these reasons that we also incorporate different restrictions that prevent excessive water use. This is of vital importance since it would allow the natural renewal of water resource sources.

All these factors make the problem computationally more interesting. Detailed information on restrictions is set out in Section 3. To test the proposal, we designed and coded a generator of feasible instances that were solved

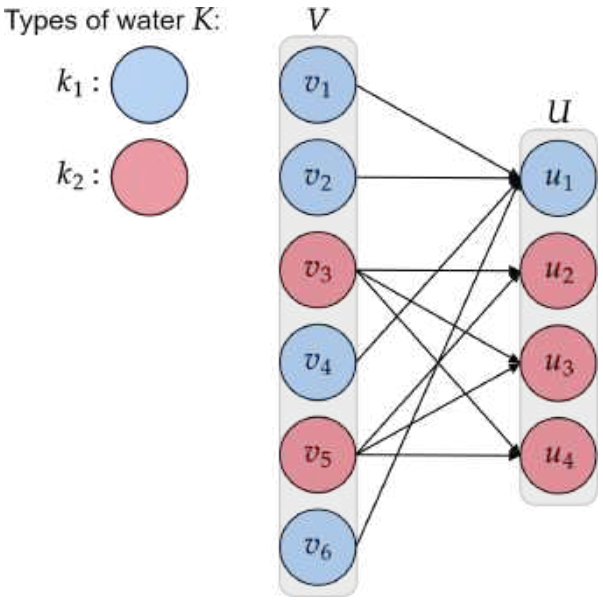


Fig. 1. Example scenario: $|V| = 6$, $|U| = 4$, $|K| = 2$ and $|C| = 3$

using the proposed mathematical model; however, due to approached restrictions, there are limits in the size of the instances that can be generated in feasible computational time.

It is important to note that the objective of the work is to show a mathematical model that allows the management of water resources considering restrictions attached to reality, so the application of metaheuristic approaches is outside the scope of this work. However, in Section 5, we establish the necessary guidelines to address the problem through metaheuristic optimization, which is why we frame it as future work.

To the best of our knowledge, there is no approach similar to the one proposed in the literature, so the results reported in the present work provide valuable knowledge to experts in the field of computational sciences and water resources management. Providing an approach that helps make informed decisions based on data.

The rest of the paper is organized as follows: Section 2 shows the background about water resources and mathematical optimization. Specifically linear programming and the transportation problem. In Section 3, the proposal is described in detail, which consists of a Mixed Integer Quadratically Constrained Program (MIQCP). This mathematical model takes as basis the classical transportation problem. Besides, the assumptions and limitations of this mathematical model are discussed.

Section 4 describes the followed methodology and the faced challenges to generate the instances. Section 5 performs an analysis of the obtained results. Finally, Section 6 states the conclusions and discusses the possible future work directions of this work.

2 Background

2.1 Water Resources

Various types of water resources originate from natural sources and serve human, agricultural, or industrial purposes. [3] classify the water resources into two main categories: surface water and groundwater. Surface water includes water flows that traverse the earth's surface (such as rivers). It encloses bodies of water gathered in naturally occurring or human-made depressions, like dams and lakes, as well as in periodically or permanently flooded areas, such as swamps and wetlands.

Groundwater consists of rainwater retained in impermeable soil. This resource holds significance as it functions both as a versatile natural water storage and a distribution network for a country. The term physical water stress refers to the ratio of water usage to available water, and it is determined by a combination of various factors [18]. The global rise in water scarcity is a consequence of escalating physical water stress, impacting regions worldwide. It's worth noting that the quality and availability of these water resources vary based on factors like geographic location, land use practices, climatic conditions, population growth, infrastructure development, over-extraction, and regulatory policies.

2.2 Water Management

As outlined in The 2030 Agenda for Sustainable Development [14] presented by the United Nations, there are 17 established Sustainable Development Goals. The sixth goal, known as SDG 6, aims to guarantee the accessibility and sustainable supervision of water and sanitation, along with the sustainable handling of water resources, water quality, integrated water resources management, water-related ecosystems, and the creation of a conducive environment.

The 2023 United Nations World Water Development Report [18] asserts that the demand for water in agriculture is primarily influenced by irrigation, with variations dependent on various determining factors. Another crucial factor to consider is the per capita water availability, which has been diminishing due to the growth rates in population. Therefore, efforts have been made to implement initiatives aimed at developing alternatives that streamline decision-making and enhance the prediction of diverse factors.

The goal is to optimize water management with greater efficiency. Models can help to represent the interactions between these factors and their complex interactions. As highlighted by [8], mathematical models are primarily categorized into two main types: simulation-based or optimization-based models. The last one can be further sub-categorized into three distinct groups: conflict resolution models, water resources planning models, and models addressing water availability and demand diagnosis.

The last category helps to estimate the water availability and compare it with the water demand to find optimal strategies for meeting these demands efficiently. Although these models can provide important information, the final decisions rest with the stakeholders.

2.3 Linear Programming (LP) and Mathematical Optimization

LP [12] or lineal optimization is a mathematical method for solving optimization problems where the objective is to optimize a linear function under constraints represented as linear equalities and inequalities.

The main objective is to find the best combination of all the variables that satisfy all the constraints for the problem to determine a way to achieve the best outcome (for example, determine the lowest cost). The following equations (1)-(6) represents the standard form of a LP [25]:

Maximize

$$c_1\,x_1 + c_2\,x_2 + \ldots + c_n\,x_n. \tag{1}$$

Subject to:

$$a_{11}\,x_1 + a_{12}\,x_2 + \ldots + a_{1n}\,x_n \leq b_1, \tag{2}$$
$$a_{21}\,x_1 + a_{22}\,x_2 + \ldots + a_{2n}\,x_n \leq b_2, \tag{3}$$
$$\vdots \tag{4}$$
$$a_{m1}\,x_1 + a_{m2}\,x_2 + \ldots + a_{mn}\,x_n \leq b_m, \tag{5}$$
$$x_1,\, x_2,\, \ldots,\, x_n \geq 0, \tag{6}$$

where the b_is, c_is, and a_{ij}s are fixed real constant numbers, and the x_is are real numbers to be determined which are called decision variables. Generally, a classical LP satisfies the following conditions: the variables of the problem must be non-negative, the objective function should express a linear combination of variables through a linear function, and the constraint set must consist of linear equations or inequalities.

The model should adapt to the problem by considering all the specific variables and constraints it must fulfill. LP has been widely used in different problems such as the routing selection problem [17], the transportation problem [21], and the supply-chain problem [22].

In addition to LP, there are other practical mathematical optimization approaches for scenarios that cannot satisfy linearity. Mixed Integer Programming (MIP) is a mathematical optimization approach based on the general principles of LP, but its decision variables consist of both integer and real values.

The classification of MIP problems depends on the nature of the objective function and constraints. The problem is called a Mixed Integer Linear Program (MILP) when the objective function and constraints are linear. However, if the objective function includes a quadratic term, it is called a Mixed Integer Quadratic Problem (MIQP).

In addition, a model is said to be a Mixed Integer Quadratically Constrained Program (MIQCP) [30] if it contains constraints with quadratic terms, regardless of the form of the objective function.

2.4 The Transportation Problem

The transportation problem stands as an essential optimization problem widely investigated in the field of operations research. Its main application lies in the efficient distribution of goods from a predefined set of source vertices to a designated set of destination vertices, with the general objective of minimizing the associated costs.

As a fundamental element in various economic, social, and market scenarios, the Transportation Problem assumes a critical role in optimizing logistics processes [6]. Formally, the Transportation Problem is stated as follows:

Consider the set of supply vertices, denoted as $V = \{v_1, v_2, \cdots v_n\}$, and a supply function $S : V \rightarrow \mathbb{R}^+$, where each vertex $v_i \in V$ is endowed with the capacity to transport up to $S(v_i)$ units of goods. Let $U = \{u_1, u_2, \cdots u_m\}$ represent the set of demand vertices corresponding to sites necessitating the delivery of goods.

The demand function is defined as $D : U \rightarrow \mathbb{R}^+$, specifying that each vertex $u_j \in U$ requires the fulfillment of a demand amounting to $D(u_j)$. The classical transportation problem can be formally characterized through Expressions (7)-(10):

$$\min \sum_{v_i \in V} \sum_{u_j \in U} c_{i,j} x_{i,j}. \tag{7}$$

Such that (s.t.):

$$\sum_{v_i \in V} x_{ij} \geq D(u_j) \forall u_j \in U, \tag{8}$$

$$\sum_{u_j \in U} x_{ij} \leq S(v_i) \forall v_i \in V, \tag{9}$$

$$x_{ij} \in \mathbb{R}^+ \forall v_i \in V, \forall u_j \in U. \tag{10}$$

The equations presented make up a Linear Programming (LP) formulation, where $c_{i,j}$ is the associated cost of transporting one unit of goods from source vertex v_i to demand vertex u_j, and x_{ij} denotes the quantity of goods units transported from v_i to u_j. Consequently, if x_{ij} goods units are transported from v_i to u_j, the corresponding cost is $c_{i,j}\, x_{i,j}$. In this LP framework, (7) is the objective function to minimize the total transportation cost from source to demand vertices.

The constraints stated in (8), ensure the satisfaction of demand for each u_j, while the (9) constraints state that the total goods shipped from the origin vertex v_i do not exceed the available quantity. Finally, the expression (10) defines the decision variables. It is important to note that this model assumes viability, that is, total supply equals or exceeds total demand, as established in the following equation:

$$\sum_{u_j \in U} D(u_j) \leq \sum_{v_i \in V} S(v_i). \tag{11}$$

It is well-known that the classical transportation problem can be solved efficiently using LP techniques [4, 5], but real-world scenarios often require more complex constraints, which pose challenges in solving these types of problems.

In the next section, we present a model based on the transportation problem that abstracts the problem of Water Resources Management. However, it presents additional restrictions that may arise in real-world scenarios, which complicate the optimization problem and increase computational demand.

3 Proposal

In this section, we propose a Mixed Integer Quadratically Constrained Program (MIQCP) specifically designed to address the Water Resources Management problem. Rooted in the fundamental principles of the transportation problem, this model incorporates additional constraints essential to address the complications inherent to the problem studied. The integration of these constraints enriches the model, which resembles real-world conditions.

3.1 Proposed Mathematical Model

In the context of water resources management, we propose to address the challenge of water distribution by modeling a scenario in which water is supplied from different water sources $V = \{v_1, v_2, ..., v_n\}$ to different demand locations $U = \{u_1, u_2, ..., u_n\}$, considering that each source and demand location manages a different type of water included in the set $K = \{k_1, k_2, \cdots, k_p\}$, where p is the number of types of water, and k_i the type of water.

In addition, we establish a supply function S, a demand function D, and a function $T : V \cup U \to K$, the latter guaranteeing that each source vertex $v_i \in V$ can supply exclusively to the demand vertices within the set $\{u_j \in U : T(u_j) = T(v_i)\}$. That is, a demand vertex that requires a specific type of water k_i, can only be satisfied by source vertices that supply the same type of water.

Since in a real context, processed water for the industry would not be sent to a place for human consumption. This delineation of water types and the associated constraints through the function T introduces an added layer of complexity to the classical transportation problem, catering to the nuanced requirements of the problem considered in this study.

In this scenario, the method of water transportation is inconsequential; That is, we ignore the specific mode of transportation and instead introduce the term "carriers", which fulfill the function of transporting water units from the source vertices to the demand vertices, establishing a cost associated with said transportation, which can be different between carriers.

This cost is a crucial factor since it is a function of all the aforementioned variables, and quantifying and optimizing it becomes essential in our study. To elaborate, we define a set of carriers, denoted as $C = \{1, 2, 3, \ldots, |C|\}$, where each carrier $l \in C$ sets an associated cost $c^l_{i,j}$ to the transport of a unit of water from a source vertex $v_i \in V$ to the demand vertex $u_j \in U$. To achieve load balancing between carriers, each operator $l \in C$ is assigned a capacity $L(l) \in \mathbb{N}$, which represents the maximum number of source vertices that it can drive.

In line with our general objective, based on this mathematical model we seek to minimize the cost of transporting water satisfying all demands. Given the multitude of constraints and variables involved, we provide a concise summary of the key assumptions underlying the problem at hand for clarity and precision:

1. All carriers can deal with any type of water, any source vertex, and any demand vertex.
2. It is established that there is sufficient capacity among carriers to operate at all origin vertices. See the following equation:

$$\sum_{l \in C} L(l) \geq |V| \qquad \text{holds.} \tag{12}$$

3. It is vitally important to consider that for each type of water, the total supply equals or exceeds the demand. This is stated in the following equation:

$$\sum_{u_j \in U : T(u_j) = k_t} D(u_j) \leq \sum_{v_i \in V : T(v_i) = k_t} S(v_i). \tag{13}$$

Holds $\forall k_t \in K$. Equations (14)-(22) introduce a mathematical model for the described problem:

$$\min \sum_{l \in C} \sum_{v_i \in V} \sum_{u_j \in U} c^l_{i,j}\, x^l_{i,j}. \tag{14}$$

Such that:

$$\sum_{l \in C} \sum_{\substack{u_j \in U : \\ T(v_i) \neq T(u_j)}} \tag{15}$$

$$x^l_{i,j} = 0 \forall v_i \in V, \tag{16}$$

$$\sum_{v_i \in V} \sum_{l \in C} y_{l,i} x^l_{i,j} \geq D(u_j) \forall u_j \in U, \tag{17}$$

$$\sum_{l \in C} \sum_{u_j \in U} x^l_{i,j} \leq S(v_i) \forall v_i \in V, \tag{18}$$

$$\sum_{v_i \in V} y_{l,i} \leq L(l) \forall l \in C, \tag{19}$$

$$x^l_{i,j} \in \mathbb{R}^+ \forall (v_i \in V, u_j \in U, l \in C), \tag{20}$$

$$y_{l,i} \in \{0, 1\} \forall l \in C, \forall v_i \in V, \tag{21}$$

where $x^l_{i,j}$ is the amount of water units to be shipped from v_i to u_j through carrier l, and:

$$y_{l,\,i} = \begin{cases} 1, & \text{if carrier } l \text{ is assigned to vertex } v_i, \\ 0, & \text{otherwise.} \end{cases} \quad (22)$$

In this model, the objective function (14) seeks to minimize transportation costs, encompassing all carriers. Constraint (16) dictates that demand vertices are exclusively supplied by source vertices with matching water types. Ensuring the satisfaction of water demands, constraint (17) plays a crucial role. To prevent excessive extraction and potential stress on water bodies, constraint (18) curtail the amount of water drawn from each source vertex to within its available capacity.

Pertinently, these constraints hold significant implications in the context of water supply. Meanwhile, constraint (19) safeguards against exceeding carrier capacities when attending to source vertices. Finally, the decision variables are defined and described through expressions (20)–(22).

3.2 Water Resources Optimization through Mathematical Optimization

To show the feasibility of the proposal, we show an example to clarify how the mathematical model works.

3.2.1 Objective Function Evaluation

Next, the process of evaluating the objective function is shown. That is, to find the values of the decision variables that optimize the function and simultaneously satisfy the constraints. First, we establish the scenario to optimize. Visually, this can be represented through a bipartite graph where the set of supply nodes V, the set of demand nodes U, the set of types of water K, and the set of carriers C are established.

It is important to remember that each carrier $l \in C$ establishes a cost $c^l_{i,j}$ for transporting a unit of water from $v_i \in V$ to $u_j \in U$, and a capacity $L(l)$ of supply nodes that it can attend. Furthermore, each scenario must satisfy (12) and (13), as well as the constraints (16)–(22) imposed on the model, this allows the problem to have a feasible solution. However, this feature makes the optimization problem difficult since a solution must be in the feasible space. Fig. 1 show an example scenario with $|V| = 6$ source vertices, $|U| = 4$ demand vertices, $|K| = 2$ types of water, and $|C| = 3$ carriers. Concerning the capacity of the carriers $L(l)$, the supply water units $S(v_i)$ and the demand water units $D(u_j)$, these are established randomly but complying with the restrictions (12), (13) and (16) to find a scenario with feasible solutions. Finally, transportation costs are also randomly assigned to a range of positive numbers.

Table 1. Water units x_{ij} obtained by LP

Carriers	Transportation cost	Water units
$l = 1$	$c^1_3, 2 = 1$	$x^1_3, 2 = 8$
	$c^1_3, 4 = 3$	$x^1_3, 4 = 7$
	$c^1_5, 3 = 3$	$x^1_5, 3 = 9$
$l = 3$	$c^3_6, 1 = 1$	$x^3_6, 1 = 9$

The complete instance of this scenario can be consulted in the link provided in the Test Instances section. To calculate the objective value using (14), the transportation costs of the instance above are used, which are summarized in Table (1), along with the water units $x^l_{i,j}$ obtained by the mathematical model to optimize the problem established in Fig. 1, for each carrier $l \in C$ and its associated transportation costs $c^l_{i,j}$. Using the values from Table (1), the objective value obtained is 65. Fig. 2 shows the optimal solution found using the mathematical model. We can verify this solution meets all established restrictions.

4 Experimental Design and Results

4.1 Test Instances

To assess the robustness of the proposed model, we formulate 30 instances that satisfy the constraints described in Section 3.1. Table (2) presents these instances along with their respective parameters, where $|V|$ and $|U|$ are the number of supply and demand vertices respectively, $|K|$ is the number of water types, and $|C|$ is the number of carriers.

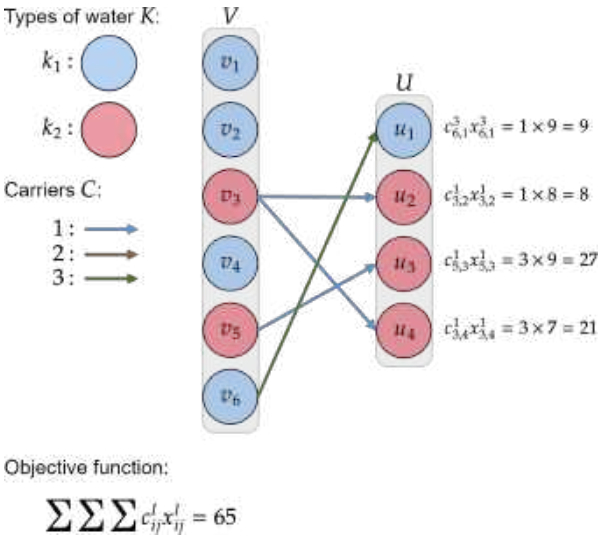


Fig. 2. Solution for the example scenario: $|V| = 6$, $|U| = 4$, $|K| = 2$, and $|C| = 3$.

The design of these instances is deliberate and features a gradual escalation of difficulty, either by adding supply and demand vertices, variations in water types, or an increase in the number of carriers. Instances 1–5 represent the simplest cases, featuring 2 types of water and 2 carriers.

The number of supply nodes $|S|$ is twice that of demand nodes $|D|$ in each instance. In contrast, Instances 6–10 mirror Instances 1–5, with the number of demand nodes $|D|$ being half that of supply nodes $|S|$. This deliberate design allows us to evaluate the model's performance under varied scenarios with unequal supply and demand nodes.

Instances 11–15 and 16–20 present a considerable increase in problem difficulty. Here, the number of water types grows by increments of 5, ranging from 5 to 25. Simultaneously, the number of carriers increases by 10 for each instance, starting at 10 and concluding at 50. Instances 11–15 and 16–20 are mirror instances, enabling a comprehensive assessment of the model's adaptability to varied configurations.

In this study, the most challenging scenarios are Instances 21–25 and 26–30, designed to push the model's limits. The complexity is heightened by increasing the number of water types by 5, from 30 to 50.

Additionally, the number of carriers increases by 50, starting at 50 and concluding at 250 for each instance. Instances 21–25 and 26–30 are mirror instances, providing a thorough exploration of the model's capabilities under difficult conditions. Finally, for these instances, water units for both supply and demand vertices were randomly assigned within the following ranges: $S(v_i) \in [1000, 5000]$, $D(u_j) \in [100, 500]$. The capacity of carriers was set randomly within the range $L(l) \in [1, |V|]$. Finally, transportation costs associated with each carrier were randomly established within the range $c^l_{i,j} \in [100, 1000]$. The complete instances can be consulted at https://github.com/alex-cornejo/WaterManagement-ComSis.

4.2 Parameter Configuration

The mathematical model was implemented in the Python programming language by using the off-the-shelf optimization software Gurobi v10. The Gurobi software implements different mathematical optimization algorithms, such as LP algorithms like Simplex and Barrier, Branch-and-Bound for MIP problems, among others [11]. All the experiments were run on a computer with a Windows 11 OS, 40 GB of RAM, and an Intel i7-10750H processor.

For the mathematical model, we tested three different relaxations available in the Gurobi software: Primal Simplex (PS), Dual Simplex (DS), and Barrier (B). Table (3) shows the results obtained from the experimentation. From this table, **OPT** refers to the optimal solutions, whereas **PS** $t(s)$, **DS** $t(s)$, and **B** $t(s)$ refer to the running time per instance for each relaxation method.

5 Analysis and Discussion of Results

In this section, we explore challenges in optimizing the proposed model with added types of water and carriers, impacting solution space, instance generation, and resolution dynamics. The obtained results showcase the potential applicability in real-world scenarios. The results reported in Table (3) affirm the feasibility of optimizing our proposal using a mathematical optimization approach. In all the cases, the optimal solutions were found.

Table 2. Test instances configuration

Instance	$\|V\|$	$\|U\|$	$\|K\|$	$\|C\|$
1	1	2	2	2
2	2	4	2	2
3	3	6	2	2
4	4	8	2	2
5	5	10	2	2
6	2	1	2	2
7	4	2	2	2
8	6	3	2	2
9	8	4	2	2
10	10	5	2	2
11	15	30	5	10
12	20	40	10	20
13	30	60	15	30
14	40	80	20	40
15	50	100	25	50
16	30	15	5	10
17	40	20	10	20
18	60	30	15	30
19	80	40	20	40
20	100	50	25	50
21	150	350	30	50
22	200	300	35	100
23	250	250	40	150
24	300	200	45	200
25	350	150	50	250
26	350	150	30	50
27	300	200	35	100
28	250	250	40	150
29	200	300	45	200
30	150	350	50	250

For this experimentation, we can appreciate that using different relaxation techniques does not change radically the running time.

However, for bigger instances, we could not ensure this. Through experimentation, we noticed that the running time of different relaxation algorithms can change drastically for some instances with $|V| > 600$. Nevertheless, we could not include experimentation for bigger instances due to the practical issues discussed below. The escalating complexity introduced by including more types of water and carriers imposes significant challenges on the problem. The imposed constraints not only shape the feasible solution space but also impact both the instance generation process and the optimization procedure.

Through empirical experimentation, we observed that the water type constraint poses a more intricate challenge for the optimization process than the carrier capacity constraint. This complexity is evident in the increased time required to resolve instances. Conversely, with a growing number of carriers, the memory requirements for processing instances also surge.

Each carrier, having an associated transportation cost expressed in a cost matrix, contributes to the memory load. For instance, if there are 200 carriers, there would be 200 distinct cost matrices per instance stored in memory. Furthermore, the size of the matrix ($|V| \times |U|$) is contingent on the number of supply and demand nodes. Therefore, spatial complexity becomes a critical consideration for both instance generation and optimization.

These challenges could be effectively addressed by adopting other optimization techniques, such as evolutionary computation of metaheuristics. For instance, solution representation in metaheuristics could involve a set of genes encoding the assignment of water types to carriers alongside other pertinent parameters. Designing crossover and mutation operators respecting problem constraints ensures the generation of feasible solutions.

Selection operators favoring diversity and exploration of the search space can be implemented. Strategies can be integrated to handle specific constraints on the type of water, such as sanctions in the objective function for non-compliance, or remedial mechanisms for infeasible solutions.

Table 3. Optimization results for each instance, reporting its optimal value **OPT** and the execution time for each strategy measured in seconds $t(s)$

Instance	OPT	PS $t(s)$	DS $t(s)$	B $t(s)$
1	235,044	0.003	0.034	0.036
2	405,326	0.001	0.002	0.000
3	656,886	0.004	0.004	0.013
4	793,611	0.003	0.004	0.004
5	489,604	0.003	0.003	0.007
6	96,280	0.002	0.001	0.001
7	227,303	0.002	0.000	0.000
8	85,908	0.004	0.005	0.004
9	249,383	0.003	0.000	0.001
10	304,674	0.004	0.004	0.006
11	1,310,305	0.027	0.022	0.081
12	1,290,357	0.034	0.031	0.050
13	2,180,535	0.081	0.086	0.083
14	2,527,523	0.162	0.146	0.167
15	3,582,503	0.270	0.297	0.283
16	705,887	0.019	0.029	0.033
17	703,887	0.022	0.016	0.017
18	1,034,468	0.068	0.062	0.066
19	1,308,086	0.140	0.159	0.150
20	1,680,449	0.296	0.303	0.299
21	10,854,584	4.454	3.864	5.518
22	9,084,413	11.084	9.272	13.205
23	7,580,286	18.027	15.151	21.556
24	6,212,880	22.829	19.389	26.393
25	4,544,234	22.190	22.589	22.874
26	4,744,219	5.067	4.297	5.902
27	5,871,442	11.502	10.102	13.886
28	7,716,824	17.815	15.057	21.407
29	9,287,937	19.743	18.116	20.121
30	10,517,914	22.349	20.405	23.261

Metaheuristics also allows for the consideration of parallelism or distribution strategies to optimize execution time, particularly crucial for large optimization problems.

Lastly, while a comprehensive study of the computational complexity of the problem would be valuable, this aspect will be rigorously addressed in future work.

6 Conclusion and Future Work

This paper introduces an innovative approach to tackling the global water stress challenge through the application of mathematical optimization, framing the Water Resources Management problem. We performed this by proposing a mathematical model, specifically an MIQCP. The proposed mathematical model is akin to the classical transportation problem, which is well-known in the field of operations research. Then, we used off-the-shelf optimization software to test the mathematical model over a set of proposed instances that consider restrictions of possible real scenarios.

The results serve as a robust affirmation, supporting the effectiveness and utility of the proposed model in addressing optimization challenges related to water use. These findings underscore the model's practical applicability and its efficacy in solving real-world problems associated with Water Resource Management optimization. Our study reveals that the inclusion of multiple water types introduces increased complexity. Instances with over 50 different water types proved more intricate, necessitating a scaling of computational resources. This adaptation becomes crucial to overcome model limitations and enhance the likelihood of finding viable solutions, hinting at the potential for specialized optimization strategies such as evolutionary computation and heuristics/metaheuristics.

In future work, we will focus on refining the model to increasingly align it with real-world scenarios, which will involve deep analysis of water management information. The complexity of the problem will also be rigorously studied, together with the possibility of improving the model by looking to linearize the constraints or propose new mathematical models with practical advantages. Finally, other strategies may be considered, particularly the implementation of evolutionary computing or heuristic/metaheuristic approaches.

Acknowledgments

We acknowledge Tecnológico Nacional de México / ITS de Purísima del Rincón for granting the necessary resources to complete this research. In addition, we would like to especially thank Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCYT) for providing the infrastructure to develop research in the country.

References

1. **Ataie-Ashtiani, B., Ketabchi, H. (2010).** Elitist continuous ant colony optimization algorithm for optimal management of coastal aquifers. Water Resources Management, Vol. 25, No. 1, pp. 165–190. DOI: 10.1007/s11269-010-9693-x.
2. **Bhavya, R., Elango, L. (2023).** Ant-inspired metaheuristic algorithms for combinatorial optimization problems in water resources management. Water, Vol. 15, No. 9, pp. 1712. DOI: 10.3390/w15091712.
3. **Castellazzi, P., Martel, R., Rivera, A., Huang, J., Pavlic, G., Calderhead, A. I., Chaussard, E., Garfias, J., Salas, J. (2016).** Groundwater depletion in central Mexico: Use of GRACE and InSAR to support water resources management. Water Resources Research, Vol. 52, No. 8, pp. 5985–6003. DOI: 10.1002/2015wr018211.
4. **Dantzig, G. B. (1951).** Application of the simplex method to a transportation problem. Activity analysis of production and allocation, Wiley, pp. 359–373.
5. **Dantzig, G. B. (1963).** Linear programming and extensions, Vol. 25. Princeton University Press. DOI: 10.1515/9781400884179.
6. **Díaz-Parra, O., Ruiz-Vanoye, J. A., Bernábe-Loranca, B., Fuentes-Penna, A., Barrera-Cámara, R. A. (2014).** A survey of transportation problems. Journal of Applied Mathematics, Vol. 2014, pp. 1–17. DOI: 10.1155/2014/848129.
7. **Ghorbani, M. K., Afshar, A., Hamidifar, H. (2021).** River water quality management using a fuzzy optimization model and the NSFWQI index. Water SA, Vol. 47, No. 1, pp. 45–53.
8. **Hernández-Cruz, A., Sandoval-Solís, S., Mendoza-Espinosa, L. G. (2022).** An overview of modeling efforts of water resources in Mexico: Challenges and opportunities. Environmental Science and Policy, Vol. 136, pp. 510–519. DOI: 10.1016/j.envsci.2022.07.005.
9. **Irani, R., Nasimi, R. (2012).** An evolving neural network using an ant colony algorithm for a permeability estimation of the reservoir. Petroleum Science and Technology, Vol. 30, No. 4, pp. 375–384. DOI: 10.1080/10916466.2010.483442.
10. **Karimlou, K., Hassani, N., Rashidi-Mehrabadi, A., Nazari, M. R. (2019).** Developing a model for decision-makers in dynamic modeling of urban water system management. Water Resources Management, Vol. 34, No. 2, pp. 481–499. DOI: 10.1007/s11269-019-02428-z.
11. **LLC, G. O. (2023).** Gurobi optimizer reference manual. www.gurobi.com.
12. **Luenberger, D. G., Ye, Y. (2021).** Linear and nonlinear programming. International Series in Operations Research and Management Science, Vol. 228. DOI: 10.1007/978-3-030-85450-8.
13. **Mohamad-Shirajuddin, T., Muhammad, N. S., Abdullah, J. (2023).** Optimization problems in water distribution systems using non-dominated sorting genetic algorithm II: An overview. Ain Shams Engineering Journal, Vol. 14, No. 4, pp. 101932. DOI: 10.1016/j.asej.2022.101932.
14. **Nations, U. (2015).** Transforming our world: The 2030 agenda for sustainable development. sdgs.un.org/2030agenda.
15. **Nguyen, D. C. H., Dandy, G. C., Maier, H. R., Ascough, J. C. (2016).** Improved ant colony optimization for optimal crop and irrigation

water allocation by incorporating domain knowledge. Journal of Water Resources Planning and Management, Vol. 142, No. 9. DOI: 10.1061/(asce)wr.1943-5452.0000662.

16. **Pahl-Wostl, C., Patterson, J. (2021).** Commentary: Transformative change in governance systems. Global Environmental Change, Vol. 71, pp. 102405. DOI: 10.1016/j.gloenvcha.2021.102405.
17. **Pascariu, B., Samà, M., Pellegrini, P., D'Ariano, A., Pacciarelli, D., Rodriguez, J. (2021).** Train routing selection problem: Ant colony optimization versus integer linear programming. IFAC-PapersOnLine, Vol. 54, No. 2, pp. 167–172. DOI: 10.1016/j.ifacol.2021.06.060.
18. **Programa mundial de la UNESCO de evaluación de los recursos hídricos (2023).** The United Nations World Water Development Report 2023: partnerships and cooperation for water. Un Water.
19. **Qin, H., Zheng, C., He, X., Refsgaard, J. C. (2019).** Analysis of water management scenarios using coupled hydrological and system dynamics modeling. Water Resources Management, Vol. 33, No. 14, pp. 4849–4863. DOI: 10.1007/s11269-019-02410-9.
20. **Radmehr, A., Bozorg-Haddad, O., Loáiciga, H. A. (2022).** Developing strategies for agricultural water management of large irrigation and drainage networks with fuzzy MCDM. Water Resources Management, Vol. 36, No. 13, pp. 4885–4912. DOI: 10.1007/s11269-022-03192-3.
21. **Rafi, F. S., Islam, S. (2020).** A comparative study of solving methods of transportation problem in linear programming problem. Journal of Advances in Mathematics and Computer Science, pp. 45–67. DOI: 10.9734/jamcs/2020/v35i530281.
22. **Sangaiah, A. K., Tirkolaee, E. B., Goli, A., Dehnavi-Arani, S. (2019).** Robust optimization and mixed-integer linear programming model for LNG supply chain planning problem. Soft Computing, Vol. 24, No. 11, pp. 7885–7905. DOI: 10.1007/s00500-019-04010-6.
23. **Seytov, A., Turayev, R., Jumamuratov, D., Kudaybergenov, A. (2021).** Mathematical models for calculation of limits in water resources management in irrigation systems. International Conference on Information Science and Communications Technologies, pp. 1–4. DOI: 10.1109/icisct52966.2021.9670304.
24. **Skardi, M. J. E., Afshar, A., Solis, S. S. (2013).** Simulation-optimization model for non-point source pollution management in watersheds: Application of cooperative game theory. KSCE Journal of Civil Engineering, Vol. 17, No. 6, pp. 1232–1240. DOI: 10.1007/s12205-013-0077-7.
25. **Vanderbei, R. J. (2020).** Linear programming: Foundations and extensions. International Series in Operations Research and Management Science, Springer, Vol. 285. DOI: 10.1007/978-3-030-39415-8.
26. **Wada, Y., Flörke, M., Hanasaki, N., Eisner, S., Fischer, G., Tramberend, S., Satoh, Y., van-Vliet, M. T. H., Yillia, P., Ringler, C., Burek, P., Wiberg, D. (2016).** Modeling global water use for the 21st century: the water futures and solutions (wfas) initiative and its approaches. Geoscientific Model Development, Vol. 9, No. 1, pp. 175–222. DOI: 10.5194/gmd-9-175-2016.
27. **Wang, H., Huang, J., Zhou, H., Deng, C., Fang, C. (2020).** Analysis of sustainable utilization of water resources based on the improved water resources ecological footprint model: A case study of Hubei province, China. Journal of Environmental Management, Vol. 262, pp. 110331. DOI: 10.1016/j.jenvman.2020.110331.
28. **Zhang, X., Chen, N., Sheng, H., Ip, C., Yang, L., Chen, Y., Sang, Z., Tadesse, T., Lim, T. P. Y., Rajabifard, A., Bueti, C., Zeng, L., Wardlow, B., Wang, S., Tang, S., Xiong, Z., Li, D., Niyogi, D. (2019).** Urban drought challenge to 2030 sustainable development

goals. Science of The Total Environment, Vol. 693, pp. 133536. DOI: 10.1016/j.scitotenv.2019.07.342.

29. **Zhu, F., Zhong, P. A., Cao, Q., Chen, J., Sun, Y., Fu, J. (2019).** A stochastic multi-criteria decision making framework for robust water resources management under uncertainty. Journal of Hydrology, Vol. 576, pp. 287–298. DOI: 10.1016/j.jhydrol.2019.06.049.

30. **Álvarez-López, J., Ceciliano-Meza, J. L., Guillén-Moya, I., Nieva-Gómez, R. (2012).** A MIQCP formulation to solve the unit commitment problem for large-scale power systems. International Journal of Electrical Power and Energy Systems, Vol. 36, No. 1, pp. 68–75. DOI: 10.1016/j.ijepes.2011.10.025.

Article received on 30/01/2024; accepted on 24/04/2024.
**Corresponding author is José Alejandro Cornejo-Acosta.*