Journal of Soft Computing and Artificial Intelligence

ISSN: 2717-8226

Journal of Soft Computing and Artificial Intelligence

Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access

Volume 05
Issue 02

December,2024

Journal of Soft Computing and Artificial Intelligence (JSCAI) is an international peer-reviewed journal that publishes integrated research articles in all areas of soft computing and artificial intelligence. The aim of the JSCAI journal is to provide a platform for researchers, professionals, and academicians around the world to combine and exchange new developments and their applications in various areas of soft computing and artificial intelligence. Journal of Soft Computing and Artificial Intelligence (JSCAI) is an international peer-reviewed journal that publishes integrated research articles in all areas of soft computing and artificial intelligence. The journal covers all branches of engineering, including mechanics, computer science, electronics, energy, aerospace engineering, materials science, nuclear engineering, systems analysis, alternative technologies, etc.

JSCAI publication, which is open access, is free of charge. There is no article submission and processing charges (APCs).

JSCAI is indexed & abstracted in:

Crossref (Doi beginning: 10.55195/jscai..xxxxxx)

Directory of Research Journals Indexing (DRJI)

Google Scholar

Index Copernicus (ICI Journal Master List)

OpenAIRE

Asos Index

Directory of Open Access scholarly Resources (ROAD)

Table of Contents

Journal of Soft Computing and Artificial Intelligence

Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access

Volume 05
Issue 02

December, 2024

Research Article

Integrated Multimedia Wireless Sensor Node for Comprehensive Lifelogging and Beyond: Design, Development and Applications

Mehmet Selim Elmalı[1]**, *Bektöre Elmalı***[2]**, *Adnan Yazıcı***[3]

[1]*Kyrgyz-Turkish Manas University, Higher School of Vocational Education, Bishkek, Kyrgyz Republic*
[2]*Gazi University, Faculty of Technology, Department of Electrical - Electronic Engineering*
[3]*Nazarbayev University, School of Engineering and Digital Sciences, Astana, Kazakhstan*

ARTICLE INFO

Article history:
Received **May 29, 2024**
Revised **September 20, 2024**
Accepted **November 25, 2024**

Keywords:
Integrated MWSN,
Environmental Monitoring,
Lifelogging,
Customizable Experiences

ABSTRACT

This paper introduces the Integrated Multimedia Wireless Sensor Node (IMWSN), a significant advancement in environmental monitoring and lifelogging within Multimedia Wireless Sensor Networks (MWSN). MWSNs, equipped with wearable sensors, are crucial for documenting personal life experiences. However, current MWSNs often lack the ability to fully integrate data across various sensor types, including environmental, visual, and medical sensors. The IMWSN addresses this gap by providing a comprehensive view of an individual's interactions and environment. The IMWSN is composed of multiple modules: a processor module that manages the overall system efficiently, a visual module designed to capture video footage of the surroundings, an environmental module that allows for real-time monitoring of environmental conditions, and a medical module dedicated to recording health-related data of individuals, a process often known as lifelogging. These components are encased in a custom-designed 3D-printed enclosure and powered by a durable 4500mAh mobile battery. System programming and monitoring are facilitated through the user-friendly Arduino IDE, making the experience accessible and customizable. Beyond its primary function in lifelogging, the IMWSN is remarkably versatile and suited for a range of applications. It can function as an action camera, assist in forest fire monitoring, support ambient assisted living environments, and monitor patients' health and daily activities rigorously. This adaptability makes the IMWSN a valuable and essential tool in fields that require extensive data collection and sophisticated analytical capabilities, highlighting its broad potential impact.

1. Introduction

The development of Multimedia Wireless Sensor Networks (MWSNs) [1], [2], has sparked a revolution in lifelogging, enabling individuals to document and immortalize their personal life journeys through wearable sensors. While MWSNs have gained significant attention for their ability to capture detailed and dynamic data, there remains a gap in achieving a fully integrated system incorporating diverse sensor types.

The Integrated Multimedia Wireless Sensor Node (IMWSN) significantly advances existing Multimedia Wireless Sensor Networks (MWSNs) by integrating diverse sensor types, including visual, environmental, and medical sensors, addressing the common limitations of traditional MWSNs. Conventional MWSNs often focus on individual data streams, such as environmental or medical sensors alone, leading to fragmented insights. In contrast, the IMWSN combines these various streams into a single

* Corresponding author
e-mail: mehmetselim.elmali@manas.edu.kg
DOI: 10.55195/jscai.1491525

system, enabling immersive data capture and continuous monitoring across multiple domains, such as environmental conditions and personal health, thereby offering a more comprehensive view of the user's environment and well-being.

To overcome these challenges of existing Multimedia Wireless Sensor Networks (MWSNs), we introduce the Integrated Multimedia Wireless Sensor Node (IMWSN). This innovative solution marks a crucial advancement in the field, designed to bridge the gap in data integration for lifelogging and beyond. By merging various sensor modules into a single, cohesive unit, the IMWSN enables the capture of rich and interconnected data streams, providing a holistic and coherent view of an individual’s life.

The IMWSN is distinct in its robust features: it includes a processor dedicated to system control, an advanced visual module for immersive data capture, an environmental module for continuous context monitoring, and a medical module for personalized health data recording. This integration allows for a deeper exploration of experiences, pattern recognition, and informed decision-making in environmental monitoring and healthcare. Beyond its primary role in lifelogging, the IMWSN serves as an action camera, aids in forest fire monitoring, and supports ambient assisted living, thus enhancing healthcare services and improving patient outcomes.

In this paper, we explore the development and potential applications of the Integrated Multimedia Wireless Sensor Node (IMWSN), emphasizing its essential role as a vital tool for lifelogging and its wider implications. We investigate the diverse range of methodologies in lifelogging, including vision-based, sensor-based, and hybrid systems, highlighting the field's expanding interest and its potential to enhance personal well-being and quality of life. This exploration underscores recent research trends in lifelogging, which demonstrate a growing commitment to advancing healthcare, personal wellness, and overall quality of life through innovative lifelogging technologies.

Despite progress in lifelogging research, many existing systems still fail to achieve a comprehensive integration of data from diverse sources, including medical, environmental, and visual sensors. Research often focuses on isolated aspects of lifelogging, such as using medical sensors for health monitoring [3], [4], [5], [6], employing vision-based technologies for patient surveillance, utilizing environmental sensors for applications like forest fire detection [7], [8] or fire detection in cities [9]. This fragmented approach typically leads to a partial depiction of lifelogging data, as studies tend to concentrate on singular sensor types.

Moreover, while some researchers have attempted to integrate multiple sensor types into their sensor nodes, they still face significant challenges in achieving a complete understanding of simultaneous activities. For instance, some nodes may include medical sensors but omit environmental or visual sensors [3], [10], whereas others might incorporate environmental sensors without including medical or visual capabilities [11], [12]. This lack of comprehensive sensor integration hampers the ability to fully capture and interpret the complex dynamics of daily life within lifelogging frameworks.

To overcome existing shortcomings and maximize the capabilities of lifelogging, this paper introduces the Integrated Multimedia Wireless Sensor Node (IMWSN). The core aim of the IMWSN is to amalgamate various data types—visual, environmental, and medical—into a cohesive unit. This approach is designed to offer a detailed and accurate portrayal of an individual's daily experiences and environment. By integrating these diverse data streams, the IMWSN significantly enhances the functionality and adaptability of lifelogging systems, setting the stage for innovative applications and breakthroughs in various fields. This paper details the development and explores the potential applications of the IMWSN, highlighting its role as a critical and transformative tool in the realm of lifelogging and beyond.

In summary, the key contributions of this study are as follows:

- The IMWSN enables lifeloggers to document their daily experiences more comprehensively and accurately, thus enhancing the functionality and relevance of lifelogging. This technology helps individuals obtain deeper insights into their daily activities and overall well-being.
- Beyond its primary use in lifelogging, the IMWSN supports a range of applications, including use as an action camera, forest fire monitoring, and healthcare surveillance. This adaptability opens up new possibilities for applying IMWSN technology across various fields, promoting progress in data collection and analytical techniques.
- This study tackles practical obstacles and provides critical insights for deploying the IMWSN in real-world scenarios. These insights are crucial for driving further innovations in lifelogging technologies.
- A comprehensive overview of the IMWSN’s architecture, components, and specifications provides an invaluable resource for researchers

and developers in lifelogging, sensor networks, and multimedia data integration. This detailed technical information aids in understanding, developing and improving integrated sensor systems.

These contributions highlight the transformative potential of the IMWSN in advancing the fields of lifelogging and related technologies.

The organization of this paper is outlined as follows: In Section 2, an exposition of background information and a comprehensive survey of the pertinent research is provided. Section 3 offers an intricate examination of the materials, design, and developmental methodologies employed in crafting our Integrated Wireless Sensor Node (IMWSN). The exploration of potential application domains for the IMWSN is encapsulated in Section 4. Section 5 presents an account of the tests and experiments conducted utilizing IMWSN. Prospective avenues for future research are discussed in Section 6. Finally, Section 7 encapsulates the conclusion of this paper.

2. Background and Related Work

In recent years, Wireless Sensor Networks (WSNs) have experienced substantial advancements, leading to the evolution of Multimedia Wireless Sensor Networks (MWSNs). These networks incorporate the capability to capture and transmit various forms of multimedia data such as images, audio, and video. MWSNs have been applied extensively in areas like healthcare, environmental monitoring, disaster management, and urban planning [13], [14]. Despite this progress, integrating multiple sensor types—particularly visual, environmental, and medical sensors—into a unified system remains a significant challenge. The Integrated Multimedia Wireless Sensor Node (IMWSN), presented in this study, is designed to address these integration challenges, enhancing the overall functionality and performance of MWSNs by adopting a holistic multi-sensor approach to data collection and analysis.

The literature reveals that many existing systems focus on integrating only one or two types of sensors—typically medical, environmental, or visual—neglecting the others. This compartmentalized approach results in incomplete data sets that do not fully capture the complexities of the monitored environment or the individual's interactions within it.

The utilization of Wireless Sensor Networks (WSNs) has witnessed a notable global surge in recent times, prompting substantial research endeavors in this realm.

2.1. Application spectrum

Ramson et al. [14] and Almalkawi et al. [13] identified a multitude of diverse application domains, ranging from agriculture to healthcare monitoring, among others. Similarly, Kandris et al. [15] delineated over 20 application areas, including Battlefield Surveillance and Water Monitoring, among others. Notably, one of the notable application spheres for WSNs is the realm of lifelogging.

In their work [16], Poon et al. delved into diverse sensor node classifications. A subset of these nodes incorporated medical sensors but lacked weather and visual sensors, whereas others integrated weather sensors but omitted medical and visual sensors.

2.2. Lifelogging

Lifelogging, the process of recording various aspects of an individual's daily life, has gained significant attention due to the rise of wearable technology and mobile sensors. Traditional lifelogging systems typically rely on singular data streams like visual or location-based data, which limits their ability to capture a complete and contextual view of the individual's health and environment.

Ninh [17] investigated stress detection by analyzing the lifelog data. In his research, he used datasets obtained from medical sensors such as heart rate and blood volume pulse but no data about environmental conditions.

In [18], Bruun et al. delved into lifelogging in natural settings, utilizing Galvanic Skin Response (GSR) sensors and a photo camera. However, their study did not encompass the tracking of environmental data, including variables such as temperature and humidity.

In [19] Cho et al. devised a lifelog-based lighting system wherein they utilized environmental and visual sensors to gather pertinent data. However, it's notable that their design did not incorporate medical data.

By integrating multiple sensor modalities such as visual, environmental, and medical data, the IMWSN offers a more holistic lifelogging approach. This integration provides deeper insights into how environmental factors and health metrics interact during daily life events

2.3. Healthcare and elderly care

In healthcare, the integration of MWSNs has become increasingly important, particularly for monitoring patients in remote or underserved areas.

Mohankumar et al. [3] expounded upon a methodology for tracking patients' health conditions. Within their study, they harnessed medical sensors to amass insights into patients' health status. Nevertheless, their approach omitted meteorological elements like atmospheric pressure, environmental temperature, humidity, light intensity, and visual data.

Sathyanarayana et al. [20] conducted a survey concerning vision-based patient monitoring systems, solely employing vision-based sensors without integrating medical or environmental counterparts.

Jeygar et al. [10] devised a sensor node featuring medical sensors such as cardiac rate and body temperature, dedicated to monitoring patients' health status. Regrettably, this node lacked environmental sensors.

Tang et al. [21] presented an intricate exploration of robot partners embedded with motion and visual sensors, accompanied by an environmental illumination sensor for elderly care. However, their endeavor omitted medical sensors.

The IMWSN enhances this potential by integrating visual, environmental, and medical data, enabling a more comprehensive approach to remote health and elderly monitoring systems.

2.4. Environmental monitoring

Environmental monitoring represents another key application for MWSNs. Tracking environmental conditions such as temperature, humidity, and pollutants in real-time has significant implications for urban planning and disaster management.

Mohapatra et al. [11] embarked on research centered on forest fire monitoring. In section III of their investigation, they introduced a sensor node model comprising environmental sensors such as temperature, humidity, smoke, and light, while omitting visual and medical sensors.

Castro-Correa et al. [7] developed a sensor node designed to monitor and detect forest fires. This node featured temperature, humidity, and gas sensors for environmental monitoring but notably lacked visual imaging capabilities and medical sensors for broader situational assessment.

Dampage et al. [8] developed a system for forest fire detection, employing a sensor node that can detect, temperature, humidity, light intensity level, and CO level. highlighting the need for integrating visual and environmental data to improve early detection systems.

The IMWSN builds on these by incorporating both environmental sensors and visual modules, providing comprehensive data for monitoring natural disasters like forest fires, floods, and extreme weather events. Additionally, the IMWSN has potential applications in urban planning, helping city planners monitor air quality and environmental conditions in real-time, thereby promoting healthier and more sustainable urban environments

2.5. Agriculture

Srbinovska et al. [12] developed a system for monitoring and controlling crop production, employing sensor nodes to detect temperature and humidity. Yet, due to the absence of visual or medical sensors, this system exhibited limitations for lifelogging purposes.

[22] Malik et al. studied wireless sensor nodes in agriculture, using sensor nodes with environmental sensors, but no visual or medical sensors.

Rajak et al. [23] explored the use of AI technology to enhance agricultural production. Their study relied on data from environmental sensors, such as temperature and humidity, but did not include medical data. However, the system's lack of visual or medical sensors limited its applicability for lifelogging purposes.

The IMWSN addresses these gaps by combining visual, environmental, and medical sensors into a unified system. This holistic approach enables the collection of comprehensive data streams, allowing for more detailed lifelogging, health monitoring, and environmental tracking. By integrating multiple sensor modalities, the IMWSN enhances the ability to recognize patterns and make informed decisions in real time, making it a valuable tool for applications that require a thorough understanding of both the environment and an individual’s health.

3. Materials, Design, Development and Applications

To effectively capture and understand the full spectrum of ongoing activities, it is essential to amalgamate various types of data, such as visual, environmental, and medical information. In response to this need, we propose the development of a specialized integrated wireless sensor node tailored specifically for lifelogging purposes. Our strategy focuses on providing a holistic view, facilitating a more detailed and accurate depiction of an individual's experiences and surroundings.

3.1. Design

Figure 1 illustrates the fundamental architecture of the system.

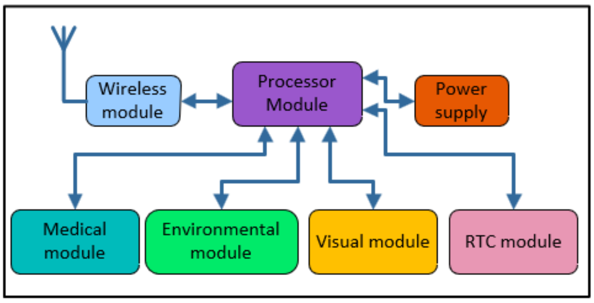


Figure 1 Basic system architecture

Communication between the processor module and other components is managed through the I2C interface. The system achieves wireless connectivity through Wi-Fi capabilities embedded within the processor module. It is powered by a durable 4500 mAh mobile phone battery. Programming of the system is conducted via a PC or notebook connected through the USB port. All components are encased in a custom-designed 3D-printed plastic enclosure. Data captured by the sensors is saved in a text file on the SD card and can also be monitored in real-time through the serial monitor connected via USB.

3.1.1. Materials

The Integrated Multimedia Wireless Sensor Node comprises a processor module, a visual module, an environmental module, a medical module, a real-time clock (RTC) module, a wireless module, a protective case, and a power supply.

3.1.2. Processor module

The processor module is the central component of the IMWSN, responsible for running the system's software and managing the operation of all other modules as shown in Fig. 2(a). It is based on the Arduino Portenta H7 [24]. which has both Arm® Cortex®-M7 and Cortex®-M4 cores for efficient data processing and management.

This module handles the communication between other components using I2C interfaces and provides power distribution to the entire system, including charging the system's battery.

Besides its robust processing capabilities, it features WiFi and Bluetooth® for efficient wireless data transmission. The module is compatible with UART, SPI, and I2C interfaces, offering broad connectivity with various peripherals. Adding to its adaptability, the module includes a versatile USB-C port. This port not only powers the board but also functions as a USB hub, can connect to a DisplayPort monitor, or provide power to other devices through OTG. Importantly, the processor module also supplies power to the other modules in the system. Its role is critical in integrating data from all other modules and ensuring that the system operates smoothly.

Processor module Specifications:

STM32H747 dual-core processor	8MB SDRAM	16MB NOR Flash
USB HS	SPI	I2C
UFL Connector (Antenna)	WiFi 802.11b/g/n 65 Mbps	Bluetooth 5.1 BR/EDR/LE
Board operating voltage 3.3 V	Board input voltage (VIN) 5 V	

3.2.2. Visual module

To monitor visual data, we utilized the Arduino Portenta Vision Shield [25], shown in Fig. 2(b). The visual module, which incorporates the Himax HM-01B0 camera, captures visual data to support both environmental monitoring and lifelogging applications. With the ability to record at various resolutions and frame rates for added flexibility, it enables the system to document the environment and life experiences in rich detail. This module is especially useful for applications like forest fire monitoring and personal lifelogging, as it can capture real-time video or images to provide critical visual context in various environments.

Visual module Specifications:

Camera: Himax HM-01B0 camera module	Supports QQVGA (160x120) at frame rates of 15, 30, 60, and 120 FPS, and QVGA (320x240) at 15, 30, and 60 FPS.
Resolution: 320 x 320 active pixel resolution with QVGA support	Image sensor: Incorporates high sensitivity 3.6µ BrightSense™ pixel technology.
Equipped with an I2C serial interface.	Ultra-low-power Image Sensor (ULPIS) engineered for Always On vision devices and applications.

3.2.3. Environmental module

For the collection of environmental data, we have employed the Arduino MKR ENV Shield rev2 [26], depicted in Fig. 2 (c). The environmental module is equipped with sensors that measure temperature, humidity, atmospheric pressure, and light intensity. This module is essential for continuous environmental monitoring, especially in applications such as forest fire detection and ambient environmental tracking for lifelogging. The real-time data collected by this module can be used to detect changes in the environment, aiding in early warning systems and providing context for lifelogging data.

Environmental module Specifications:

Power Supply Voltage: 3.3V/5V	Operational Current: Less than 15mA
Communication Method: I2C/UART	

Figure 2 (a) Arduino Portenta H7 board (b) Arduino Portenta Vision Shield board (c) ENV Shield board

3.2.4. Medical module

For the collection of medical data, we employed the Gravity: MAX30102 PPG Heart Rate and Oximeter Sensor board [27], depicted in Fig. 3. (a). The medical module, featuring a heart rate and oxygen saturation sensor (MAX30102), allows the IMWSN to track health-related data. This module is particularly important for lifelogging applications that involve personal health monitoring, such as tracking vital signs in ambient assisted living environments or during physical activities. By integrating health data, the IMWSN goes beyond environmental monitoring, offering a comprehensive view of both the user's environment and health.

3.2.5. RTC module

For timestamping the collected data, we utilized the DS3231 RTC (Real-Time Clock) [28] module, depicted in Fig. 3. (b). Its continuous operation is ensured by a CR2032 battery, guaranteeing the preservation of time information even when the IMWSN is powered off. The RTC module ensures accurate timestamping of all collected data, which is crucial for synchronizing data from multiple sources, especially in lifelogging scenarios. This enables the system to maintain precise records of when environmental or health events occur, further enhancing the usefulness of the collected data.

RTC module Specifications:

Serial Protocol: I2C	I2C Address: 0x68
Operating Voltage: 5V or 3.3V DC (External)	CR2032 clock battery slot
Clock Precision: ±1 minute per year	Dimensions: 38x22x14mm

Medical sensor module specifications:

Power Supply Voltage: 3.3V/5V	Operational Current: Less than 15mA
Communication Method: I2C/UART	I2C Address: 0x57

3.2.6. Wireless module

A wireless module is embedded into the processor board. It provides Wi-Fi and Bluetooth® connectivity options for seamless data exchange to the system.

3.2.7. Power supply

As the power supply, we utilized a 4500 mAh mobile phone battery, shown in Fig. 3 (c). The Arduino Portenta H7 board charges this power supply through its USB port. The inclusion of a 4500mAh mobile battery ensures that the IMWSN is fully mobile, allowing it to function independently of fixed power. This mobility is particularly important in applications that require movement, such as when the IMWSN is mounted on a moving object or individual for lifelogging, or in remote areas where continuous environmental monitoring is required without a direct power supply. The battery's capacity enables long-term use, making the IMWSN suitable for extended operations in scenarios like forest fire monitoring or personal health tracking, where frequent recharging would be impractical.

Figure 3 (a) Heart Rate and Oximeter Sensor board (b) DS3231 RTC module (c) Power supply

3.2.8. Case

The IMWSN’s components are housed in a custom-designed, 3D-printed plastic enclosure that provides physical protection and enhances the durability of the system as illustrated in Fig. 4. The enclosure is strategically designed with circular and rectangular apertures that align with the sensors and USB port, ensuring that all modules remain securely positioned and easily accessible for data collection. The 3D printing process allows for a lightweight yet sturdy design, which can be tailored to specific needs and applications, making it an adaptable and customizable solution for different environments. Additionally, the enclosure protects the system from external damage, dust, and other environmental factors, making it suitable for outdoor and harsh conditions such as forest monitoring or lifelogging in dynamic environments.

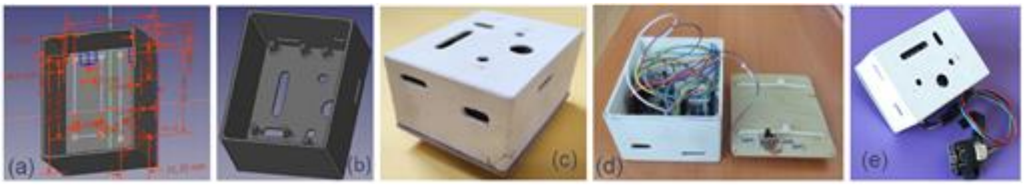


Figure 4 (a) Design with dimensions (b) 3-D view without dimensions (c) Design printed with white plastic (d) Modules installed in the case (e) Heart Rate and Oximeter Sensor connected.

3.2. Implementation

The IMWSN's modules are securely affixed within a custom-designed, 3D-printed plastic enclosure. Circular and rectangular apertures are strategically placed to align with the sensor positions and USB port, respectively, on the case's surfaces. The power supply is mounted on the inner side of the cover. To accurately timestamp acquired data, we incorporated the DS3231 RTC (Real-Time Clock) module.

We established connections between these sensor boards RTC board and the processor board using the I2C interface.

For programming the IMWSN, we employed the Arduino IDE [29], as displayed in Fig. 5. To facilitate programming and real-time monitoring, the IWSN is linked to a computer via a USB port.

The Arduino IDE provides a user-friendly editor where users can write, modify, and debug the IMWSN’s program using the Arduino programming language. This simplifies the development process, especially for those who are not experts in embedded systems. The IDE’s easy-to-use interface makes it accessible to a wide range of users, from beginners to professionals, facilitating quick iteration and updates to the system code. The Serial Monitor feature is particularly useful, allowing users to view real-time data streams from the IMWSN, which aids in debugging and monitoring the system’s performance.

The Arduino IDE efficiently compiles code and uploads it directly to the IMWSN. The process is streamlined, with built-in support for the Arduino Portenta H7 board used in the IMWSN, eliminating the need for complex configurations. This makes it easier to integrate custom sensor data, modify system behavior, or add new functionalities without needing specialized tools.

Captured sensor data is stored in a text file generated on the SD card within the visual module. Captured pictures’ names are timestamped with time data and stored in the SD card. Environmental and medical sensor data are time stamped and recorded into a .txt file which is created in the SD card.

Figure 5 Arduino IDE.

4. Possible Application Areas

While our primary goal revolved around the development of the Integrated Sensor Node for lifelogging, its scope of applications reaches far beyond this initial intention, rendering it highly adaptable for implementation in diverse domains.

4.1. Monitoring Forest Fire

The IMWSN’s ability to integrate visual, environmental, and real-time data collection makes it an ideal tool for forest fire monitoring.

Equipped with temperature, humidity, and visual sensors, it can detect early signs of fire hazards by continuously monitoring environmental conditions and capturing visual data from remote locations. Its portability and battery-powered mobility allow it to be deployed in hard-to-reach areas where conventional monitoring systems are impractical. The combination of visual feedback with environmental readings provides a comprehensive data stream, aiding in quicker detection and response to fire threats.

By proficiently observing and recording fluctuations in forest temperature, humidity, and light intensity, the IWSN gathers real-time data crucial for the early detection of fire-prone situations. Leveraging machine learning models designed for fire prediction, the IWSN possesses the capacity to identify anomalies indicative of forest fire initiation. Once identified, the IWSN can promptly trigger a pre-configured response protocol, efficiently transmitting fire alarms directly to the local fire department or relevant authorities. This proactive approach, empowered by the IWSN's advanced monitoring capabilities and rapid alerting mechanisms, plays an instrumental role in enhancing forest fire prevention and mitigation strategies, ultimately contributing to safeguarding vast ecosystems and minimizing potential damage.

4.2. Ambient Assisted Living (AAL)

In the context of ambient assisted living [30], [31], [32], the IMWSN can play a vital role in enhancing the safety and well-being of elderly or disabled individuals. By integrating medical sensors that track vital signs such as heart rate and oxygen levels, the system can continuously monitor the health status of users and detect abnormal conditions. The environmental sensors further add to this by monitoring factors like room temperature and air quality, ensuring a safe and comfortable living environment. IMWSN’s ability to wirelessly transmit data to caregivers or health professionals makes it a valuable tool for real-time health monitoring in assisted living facilities.

4.3. Patient Health Monitoring

Beyond lifelogging, the IMWSN has significant applications in patient health monitoring.

The inherent capabilities of the Integrated Multimedia Wireless Sensor Node (IMWSN), encompassing a specialized oxygen saturation level and heart rate sensor, render it a well-suited tool for the comprehensive monitoring of the health conditions of the patients.

This is especially useful in remote patient monitoring, where individuals with chronic conditions can be monitored in real-time without needing to visit healthcare facilities frequently. By integrating this data with environmental factors, the IMWSN can provide a comprehensive picture of the patient’s health, helping healthcare providers make better-informed decisions

This heightened functionality proves particularly invaluable within scenarios characterized by heightened health concerns, such as the ongoing COVID-19 pandemic. In such circumstances, the IMWSN exhibits the potential to emerge as a critical asset for healthcare providers and medical practitioners. By seamlessly and continuously measuring and tracking essential physiological parameters like oxygen saturation levels and heart rates, this technology furnishes a real-time understanding of patients' vital signs. The utility of the IMWSN becomes especially pronounced in situations where remote patient monitoring is imperative, allowing healthcare professionals to remotely assess patients' health status without requiring their physical presence. This not only minimizes exposure risks for both patients and medical staff but also augments the capacity of healthcare systems to efficiently manage patient care and allocate resources. In the context of a pandemic, the IMWSN can contribute significantly to the early identification of deteriorating health conditions, facilitating prompt interventions and potentially curbing adverse outcomes. Its multifaceted potential extends beyond the immediate pandemic response, positioning it as a versatile asset for transforming healthcare delivery and remote patient monitoring practices in the broader healthcare landscape.

4.4. Action Cameras and Other Applications

The IMWSN can also function as a mobile action camera, recording visual data in dynamic environments. Its lightweight, portable design makes it ideal for capturing adventure sports, wildlife observation, or any application requiring mobile video recording. Furthermore, its integration of environmental and health sensors makes it applicable in various industrial and research settings, where environmental monitoring or safety tracking is required.

For example, the IMWSN can seamlessly integrate with a skier's helmet, effectively transforming it into a valuable action camera and health monitoring device. With a sophisticated array of sensors, including those for ambient temperature, heart rate, and blood oxygen levels, the IMWSN continuously tracks essential physiological indicators. This unique combination of features positions the IMWSN as a critical guardian of skiers' well-being, enhancing safety and knowledge on the slopes. Through meticulous monitoring of vital parameters and swift detection of deviations from normal values, the IMWSN takes a proactive approach to risk reduction. This capability allows it to identify irregularities promptly, particularly in relation to potential heart-related issues. In essence, the IMWSN represents the fusion of adventure and innovation, emphasizing the importance of incorporating technology into recreational activities to ensure both thrilling experiences and individual health.

5. Experiments of IWMSN for Validation

The IMWSN is presently in the prototype phase, representing an early stage of development focused on foundational validation. A carefully selected set of experiments has been designed to assess and confirm the system's basic functionality. These initial tests aim to ensure the system operates as intended and provides a reliable basis for future enhancements.

The developed Integrated Sensor Node prototype was tested by conducting measurements on volunteer participants and at meteorological stations, yielding the following findings. The data collected from these measurements was saved to a text file on the SD card. A snapshot of this text file is shown in Fig. 6 (c).

5.1. Measurements Conducted with the Medical Module of the Integrated Sensor Node Prototype

A comprehensive evaluation of the medical module's efficacy was conducted, encompassing a cohort of sixteen participants, comprising three male and three female volunteers. The diverse composition of the study group facilitates a more encompassing understanding of the module's performance across varying physiological profiles. The culmination of this testing effort culminates in the detailed documentation of SPO2 and heart rate data, meticulously cataloged in Table 1, Table 2 and Table 3.

The measurements are taken by devices produced by "Shanghai Berry" and "IMDK Medical" companies and the Integrated Sensor Node prototype This repository of data encapsulates the responses of the participants to the medical module's probing, offering insights into individual physiological characteristics and responses.

These measurements provide a focused snapshot that demonstrates the module's precision in acquiring and recording essential health metrics. This deliberate approach underscores the module's potential for personalized health monitoring and emphasizes its role in contributing to individualized healthcare solutions.

Table 1 Measurement values taken by the medical module.

1st participant (Female)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
26.04.2024	16:41	98	64	99	65	99	65
29.04.2024	09:22	98	71	99	70	99	70
29.04.2024	11:30	98	62	99	64	99	63
30.04.2024	14:43	98	71	99	72	99	75

2nd participant (Female)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
29.04.2024	09:30	98	65	98	64	99	63
29.04.2024	11:27	100	61	99	65	99	60
30.04.2024	14:15	98	64	99	67	99	61
30.04.2024	16:55	98	60	99	60	99	58

3rd participant (Female)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
26.04.2024	16:59	97	70	99	71	99	73
29.04.2024	12:54	99	82	99	78	99	79
30.04.2024	09:43	98	79	98	76	99	81
30.04.2024	14:03	98	78	97	78	99	80

4th participant (Male)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
26.04.2024	16:04	98	64	99	62	99	62
29.04.2024	09:42	97	81	99	84	99	83
29.04.2024	11:18	96	78	97	80	98	77
30.04.2024	14:51	97	87	98	82	99	85

5th participant (Male)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
26.04.2024	16:10	98	70	98	74	99	67
29.04.2024	09:26	97	75	98	77	99	74
29.04.2024	11:34	97	68	99	70	99	67
30.04.2024	15:25	98	81	98	77	99	78

6th participant (Male)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
26.04.2024	16:48	97	87	97	86	99	87
29.04.2024	09:34	99	93	98	89	99	92
29.04.2024	13:37	99	108	98	109	99	105

30.04.2024	14:56	97	92	97	90	99	87

7th participant (Male)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
13.10.2024	16:13	95	86	95	85	97	85
24.10.2024	09:14	98	82	99	84	98	81
24.10.2024	15:30	97	85	96	84	98	84
28.10.2024	14:55	97	84	97	85	97	85

8th participant (Male)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
23.10.2024	16:23	97	77	99	78	99	76
24.10.2024	15:25	99	82	98	82	99	80
25.10.2024	10:44	99	70	99	72	99	68
28.10.2024	14:45	97	81	99	85	99	77

9th participant (Female)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
23.10.2024	16:37	98	79	99	77	99	79
24.10.2024	09:25	99	96	98	97	99	94
24.10.2024	15:36	99	86	99	85	99	85
25.10.2024	11:13	99	86	97	88	99	85

10th participant (Female)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
23.10.2024	16:45	98	79	99	78	97	80
24.10.2024	09:30	98	89	99	88	99	89
24.10.2024	15:59	98	90	99	91	99	88
25.10.2024	11:01	99	85	99	83	99	87

11th participant (Male)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
23.10.2024	16:55	98	59	99	61	99	58
24.10.2024	09:40	98	60	98	58	99	62
24.10.2024	16:06	98	66	97	65	99	65
25.10.2024	10:51	98	68	98	67	99	70

12th participant (Female)							
		IWSN		Shanghai Berry		IMDK Medical	
Date	Time	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)	SPO2 (%)	Heartbeat (bpm)
23.10.2024	17:10	97	74	99	71	99	73
24.10.2024	11:30	98	72	99	69	99	70
24.10.2024	16:16	98	74	99	76	99	72
25.10.2024	10:35	97	75	99	74	99	76

Table 2 The differences in the measurement values taken by the medical module

1st participant (Female)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
1.01	1.01	1.54	1.54
1.01	1.01	-1.43	-1.43
1.01	1.01	3.13	1.59
1.01	1.01	1.39	5.33

2nd participant (Female)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
0.00	1.02	-1.56	-3.17
-1.01	-1.01	6.15	-1.67
1.01	1.01	4.48	-4.92
1.01	1.01	0.00	-3.45

3rd participant (Female)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
2.02	2.02	1.41	4.11
0.00	0.00	-5.13	-3.80
0.00	1.02	-3.95	2.47
-1.03	1.03	0.00	2.50

4th participant (Male)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
1.01	1.01	-3.23	-3.23
2.02	2.02	3.57	2.41
1.03	2.06	2.50	-1.30
1.02	2.04	-6.10	-2.35

5th participant (Male)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
0.00	1.02	5.41	-4.48
1.02	2.04	2.60	-1.35
2.02	2.02	2.86	-1.49
0.00	1.02	-5.19	-3.85

6th participant (Male)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
0.00	2.06	-1.16	0.00
-1.02	0.00	-4.49	-1.09
-1.02	0.00	0.92	-2.86
0.00	2.06	-2.22	-5.75

7th participant (Male)			
SPO2 (%) difference		Heartbeat difference	

Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
0.00	2.11	-1.18	-1.18
1.01	0.00	2.38	-1.23
-1.04	1.03	-1.19	-1.19
0.00	0.00	1.18	1.18

8th participant (Male)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
2.02	2.06	1.28	-1.32
-1.02	0.00	0.00	-2.50
0.00	0.00	2.78	-2.94
2.02	2.06	4.71	-5.19

9th participant (Female)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
1.01	1.02	-2.60	0.00
-1.02	0.00	1.03	-2.13
0.00	0.00	-1.18	-1.18
-2.06	0.00	2.27	-1.18

10th participant (Female)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
1.01	-1.02	-1.28	1.25
1.01	1.02	-1.14	0.00
1.01	1.02	1.10	-2.27
0.00	0.00	-2.41	2.30
11th participant (Male)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
1.01	1.02	3.28	-1.72
0.00	1.02	-3.45	3.23
-1.03	1.02	-1.54	-1.54
0.00	1.02	-1.49	2.86

12th participant (Female)			
SPO2 (%) difference		Heartbeat difference	
Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
2.02	2.06	-4.23	-1.37
1.01	1.02	-4.35	-2.86
1.01	1.02	2.63	-2.78
2.02	2.06	-1.35	1.32

Table 3 Results of statistical analysis of the differences in the measurements taken by the medical module.

	SPO2 (%) difference		Heartbeat difference	
	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)	Difference from Shanghai Berry (%)	Difference from IMDK Medical (%)
Minimum difference	-2.06	-1.02	-6.10	-5.75
Maximum difference	2.02	2.11	6.15	5.33

Average	0.46	0.99	-0.07	-0.97
Standard deviation	1.02	0.84	2.97	2.50

The average difference in SPO2 measurements between IWSN and Shanghai Berry is 0.46, with a standard deviation of 1.02. This variance is considered acceptable.

For IWSN and IMDK, the average SPO2 difference is 0.99, with a standard deviation of 0.84, which is also deemed reasonable. The average difference between IWSN and Shanghai Berry for heartbeat measurements is -0.07, with a standard deviation of 2.97. The average difference between IWSN and IMDK for heartbeat measurements is -0,097, with a standard deviation of 2.50. Although the variations in heartbeat measurements are larger than those in SPO2, these are considered reasonable, as heart rate can fluctuate rapidly and momentarily.

5.2. Measurements Conducted with the Environmental Module of the Integrated Sensor Node Prototype

The data acquired from rigorous testing of the environmental module across various times of the day and various days is meticulously compiled and presented in Table 4 and Table 5.

Measurements were conducted with the environmental module of the Integrated Sensor Node and the instruments of "Central-Asian Institute for Applied Geosciences (CAIAG)" at the measurement station of CAIAG located in the village of Kashkasuu, Bishkek.

Each dataset created as the result of measurements captures a snapshot of conditions within this station, revealing dynamic variations in temperature, humidity and pressure throughout the day. The comprehensive nature of the collected data reflects the module's aptitude for portraying the nuanced environmental fluctuations inherent to real-world scenarios.

Table 4 Measurement values taken by the environmental module.

		IWSN			CAIAG		
Date	Time	Relative Humidity (%)	Temperature (C˚)	Pressure (kPa)	Relative Humidity (%)	Temperature (C˚)	Pressure (kPa)
28.04.2024	12:26	63.70	8.13	89.37	65.00	7.90	93.05
28.04.2024	13:23	62.40	8.40	91.41	63.60	8.20	93.08
28.04.2024	14:15	62.00	8.40	91.47	63.40	8.60	93.43
28.04.2024	15:06	59.00	9.50	91.50	61.60	9.00	93.14
29.04.2024	15:29	46.60	16.70	90.63	47.40	16.30	93.07
29.04.2024	16:21	42.50	17.10	91.65	44.60	16.90	93.06
29.04.2024	17:13	46.20	16.10	89.68	48.10	16.50	93.09

Table 5 The differences in the measurement values taken by the environmental module and its statistical analysis.

	Relative Humidity difference from CAIAG measurements (%)	Temperature difference from CIAG measurements (%)	Pressure difference from CAIAG measurements (%)
	2.00	-2.91	3.95
	1.89	-2.44	1.79
	2.21	2.33	2.10
	4.22	-5.56	1.76
	1.69	-2.45	2.62
	4.71	-1.18	1.52
	3.95	2.42	3.66
Minimum difference	1.69	-5.56	1.52
Maximum difference	4.71	2.42	3.95
Average difference	2.95	-1.40	2.49
Standard deviation	1.19	2.68	0.90

The average difference in relative humidity measurements between IWSN and CAIAG is 2.95 with a standard deviation of 1.19, which is considered acceptable.

For temperature measurements, the average difference is -1.40 with a standard deviation of 2.68, also within a reasonable range.

The average difference in pressure measurements is 2.49 with a standard deviation of 0.90, which we also consider reasonable.

5.3. Visual module

The visual module's capabilities are exemplified through an illustrative snapshot, showcased in Fig. 6. (a). captured with a resolution of 320X240 pixels, in alignment with the module's camera specifications. This exemplar effectively underscores the visual module's proficiency in documenting surroundings, underscoring its potential across various applications that demand accurate visual documentation. A snapshot of .bmp files recorded into the SD card is shown in Fig. 6. (b).

Together, the collated environmental data and vivid photographic portrayal illuminate the empirical prowess of the IMWSN's constituent modules, validating their roles in capturing diverse facets of the environment and imagery.

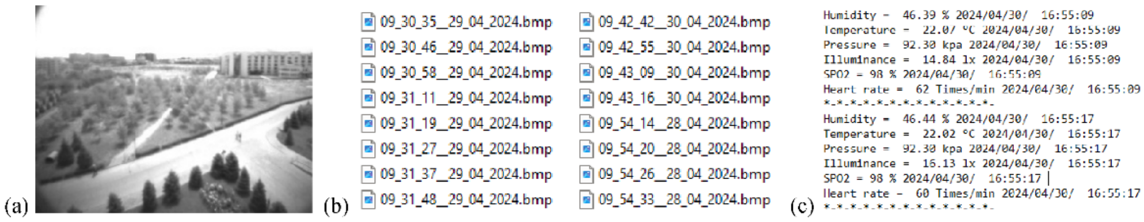


Figure 6 (a) A sample photo taken by IWSN. (b) A snapshot of .bmp files in the SD card (c) A snapshot of the recorded data in the text file.

6. Discussion and Future Directions

The research presented in this article represents a significant advancement in the field of wireless multimedia sensor networks (MWSNs) and lifelogging. The Integrated Wireless Media Sensor Node (IMWSN) addresses many of the longstanding integration challenges that have limited existing MWSN systems. By incorporating visual, environmental, and medical sensors, the IMWSN enables comprehensive, real-time data collection from multiple sources, offering transformative possibilities in a range of applications. This paper not only explores the development and implementation of the IMWSN but also highlights its wide-reaching implications, diverse applications, and exciting avenues for future research and innovation.

6.1. Transformative Impact on Lifelogging

The IMWSN represents a paradigm shift in lifelogging, moving beyond the traditional visual or location-based data to include environmental and physiological metrics. Current lifelogging systems often rely on singular data streams (e.g., location or activity) to document personal histories. However, the IMWSN's multi-sensor integration can vastly improve how individuals understand and interact with their life narratives. By capturing synchronized visual, environmental, and health data, this system enhances the depth of autobiographical memory, providing a more nuanced understanding of how an individual's health and surroundings interplay during significant life events. Such integration could enable deeper insights into personal health patterns, environmental triggers, and lifestyle impacts, thus pushing lifelogging beyond simple life documentation toward personalized health and environmental monitoring [17].

This shift in lifelogging practices could also be valuable for researchers in psychology and healthcare. By monitoring both environmental and physiological data, the IMWSN can provide more accurate insights into how people's environments influence their mental and physical well-being. For example, studies have shown that environmental factors like temperature, air quality, and light exposure have significant effects on both mental and physical health [33]. By logging these factors, individuals and healthcare providers could identify environmental stressors and mitigate their impact on well-being.

6.2. Versatility in Critical Applications

The versatility of the IMWSN extends beyond lifelogging into critical real-world applications, including healthcare, environmental monitoring, and urban planning. Its ability to act as both an action camera and a health monitoring device makes it applicable in a wide variety of settings, from urban areas to remote environments. For example, in forest fire surveillance, the IMWSN could provide real-time data about environmental conditions such as temperature, humidity, and smoke levels. This would enable early detection of fire hazards and allow authorities to respond quickly and effectively, potentially reducing the damage caused by wildfires [11].

In healthcare, particularly during pandemics or for patients in remote or underserved areas, the IMWSN can be used to monitor critical health metrics like heart rate, body temperature, and oxygen saturation levels. This would provide real-time health data that could be transmitted to healthcare providers, enabling remote diagnosis and treatment. With the increasing focus on telemedicine and remote health monitoring, especially during global health crises like COVID-19, systems like the IMWSN can be a vital tool in ensuring healthcare access and responsiveness [4].

6.3. Broader Societal Impacts

The IMWSN's applications are not limited to individual use but extend to societal benefits in fields such as urban planning and elderly care:

- Urban Planning: By leveraging the IMWSN's comprehensive data collection capabilities, city planners could gather real-time information on environmental factors such as air quality, temperature, and humidity, and correlate these with health metrics of the population. This data could be used to create healthier, more sustainable urban environments, making cities more livable and better equipped to handle public health challenges [34].
- Elderly Care: In ambient assisted living (AAL) environments, the IMWSN could monitor the health and environmental conditions of elderly individuals, providing caregivers with real-time insights into their well-being. By continuously tracking vital signs and environmental factors like room temperature or air quality, the IMWSN could help caregivers detect health issues early and provide timely intervention, potentially improving the quality of life for elderly individuals [4], [35], [36].

6.4. Future Research and Development

While the IMWSN represents a major advancement, there are several areas that warrant further research and development to enhance its functionality:

- Sensor Expansion: Currently, the IMWSN integrates key visual, environmental, and medical sensors, but its capabilities could be expanded by integrating additional sensors [23]. For instance, motion trackers, air quality monitors [35], [37], or thermal imaging systems could provide even richer data, making the IMWSN applicable to a wider range of use cases. For example, motion sensors could be particularly useful in tracking physical activity in healthcare settings, while advanced vision systems could improve detection in search and rescue operations.
- Advanced Data Analytics [38]: The IMWSN collects large amounts of complex, multi-modal data, which requires sophisticated analytics to process and derive meaningful insights. Future work should focus on developing machine learning algorithms capable of analyzing and interpreting these complex datasets in real-time. Such algorithms could, for example, predict health risks based on patterns in the collected data or identify environmental hazards like approaching wildfires.
- Data Security and Privacy [39] Given the sensitive nature of the data collected by the IMWSN, ensuring data security and privacy is critical. Future research should focus on developing robust encryption protocols and secure data transmission techniques. Privacy-preserving methods, such as federated learning, could allow for the development of predictive models without exposing individual data, thus protecting users' privacy while still enabling the system to learn from vast amounts of collected data.
- Energy Efficiency and Sustainability: We have included a 4500mAh battery in the design allows the IMWSN to operate independently and for extended periods, which offers practical advantages over systems that rely on fixed or less portable power sources. However, prolonged operation in remote environments or for extended health monitoring requires energy-efficient hardware. Further research into low-power sensors and energy-harvesting technologies could significantly extend the

operational life of the IMWSN, making it more practical for long-term use in applications such as environmental monitoring or patient care.

- Testing the IMWSN under real-world conditions such as extended use, various weather environments, and potential sensor degradation.
- Optimizing the system for durability in real-world applications, such as enhancing battery life for prolonged use, and ensuring sensor longevity.
- Validating the system's applicability and reliability in field conditions, where performance is critical.

7. Conclusion

The IMWSN represents a major advancement in the field of MWSNs, addressing key challenges related to data integration, real-time monitoring, and application versatility. By combining visual, environmental, and medical sensors, the IMWSN enhances the richness of data available for applications ranging from lifelogging to disaster management and healthcare. Its ability to integrate data across multiple modalities provides users with a comprehensive understanding of their environment and health, enabling more informed decision-making and timely interventions.

Traditional Multimedia Wireless Sensor Networks (MWSNs) struggle with the integration of disparate data sources such as environmental signals, visual inputs, and medical metrics. These isolated data streams offer only piecemeal views of an individual's daily life, substantially limiting the depth and utility of lifelogging. Furthermore, the lack of a cohesive framework for data aggregation compromises the effectiveness of existing lifelogging systems.

In response to these limitations, we introduce the Integrated Multimedia Wireless Sensor Node (IMWSN) as a groundbreaking solution. By merging various sensor modules into a single unified device, the IMWSN facilitates the collection of rich, interconnected data streams. This innovative technology not only bridges the gap in data collection but also transforms how life experiences are comprehensively captured and interpreted.

The IMWSN marks a major leap forward in the field of comprehensive data acquisition, especially designed for lifelogging and related applications. It brings together multiple sensor modules into a unified system that adeptly captures and synthesizes complex data streams, offering a detailed and holistic view of an individual's life narrative. This paper details practical implementations and insights concerning the deployment of the IMWSN, pushing forward the development of lifelogging technologies.

Additionally, an in-depth discussion of the IMWSN's design and technical specifics serves as an invaluable resource for professionals involved in lifelogging, sensor networks, and multimedia data integration. This interdisciplinary study bridges the gaps between lifelogging, sensor technology, and data integration, fostering cross-disciplinary collaboration and encouraging researchers to pursue innovative methods for integrating diverse sensor data across various applications.

As technology continues to evolve, the IMWSN has the potential to play an increasingly important role in critical societal applications, offering solutions that can enhance both individual well-being and collective societal outcomes. Future research should focus on expanding the system's capabilities by incorporating additional sensors, improving data analytics, ensuring data security and improving energy efficiency.

References

[1] M. S. Obaidat and S. Misra, *Principles of Wireless Sensor Networks*, 1st ed. Cambridge University Press, 2014. doi: 10.1017/CBO9781139030960.

[2] M. Majid *et al.*, "Applications of Wireless Sensor Networks and Internet of Things Frameworks in the Industry Revolution 4.0: A Systematic Literature Review," *Sensors*, vol. 22, no. 6, Art. no. 6, Jan. 2022, doi: 10.3390/s22062087.

[3] M. Mohankumar, Kirthana, P. Banu, M. Shree, and M. Mylsamy, "MULTI-PARAMETER SMART HEALTH MONITORING SYSTEM USING ARDUINO-UNO," pp. 2582–5208, Feb. 2022.

[4] Y. Yang, H. Wang, R. Jiang, X. Guo, J. Cheng, and Y. Chen, "A Review of IoT-Enabled Mobile Healthcare: Technologies, Challenges, and Future Trends," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 9478–9502, Jun. 2022, doi: 10.1109/JIOT.2022.3144400.

[5] M. Sarkar, T.-H. Lee, and P. K. Sahoo, "Smart Healthcare: Exploring the Internet of Medical Things with Ambient Intelligence," *Electronics*, vol. 13, no. 12, Art. no. 12, Jan. 2024, doi: 10.3390/electronics13122309.

[6] T. Jabeen *et al.*, "Smart Wireless Sensor Technology for Healthcare Monitoring System Using Cognitive Radio Networks," *Sensors*, vol. 23, no. 13, Art. no. 13, Jan. 2023, doi: 10.3390/s23136104.

[7] J. A. Castro Correa, S. B. Sepúlveda Mora, B. Medina Delgado, C. D. Escobar Amado, and D. Guevara Ibarra, "A forest fire monitoring and detection system based on wireless sensor networks," *Sci. tech*, vol. 27, no. 2, pp. 89–96, Jun. 2022, doi: 10.22517/23447214.24784.

[8] U. Dampage, L. Bandaranayake, R. Wanasinghe, K. Kottahachchi, and B. Jayasanka, "Forest fire detection system using wireless sensor networks and machine learning," *Sci Rep*, vol. 12, no. 1, Art. no. 1, Jan. 2022, doi: 10.1038/s41598-021-03882-9.

[9] F. M. Talaat and H. ZainEldin, "An improved fire detection approach based on YOLO-v8 for smart cities," *Neural Comput & Applic*, vol. 35, no. 28, pp. 20939–20954, Oct. 2023, doi: 10.1007/s00521-023-08809-1.

[10] V. Noel Jeygar Robert, P. Ragupathy, K. Chandraprabha, A. S. Nandhini, and M. Gnanasekaran, "Multi-Parameter Smart Health Monitoring System using Internet of Things," in *2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, Feb. 2022, pp. 1326–1334. doi: 10.1109/ICAIS53314.2022.9742828.

[11] S. Mohapatra and P. M. Khilar, "Forest fire monitoring and detection of faulty nodes using wireless sensor network," in *2016 IEEE Region 10 Conference (TENCON)*, Nov. 2016, pp. 3232–3236. doi: 10.1109/TENCON.2016.7848647.

[12] M. Srbinovska, C. Gavrovski, V. Dimcev, A. Krkoleva, and V. Borozan, "Environmental parameters monitoring in precision agriculture using wireless sensor networks," *Journal of Cleaner Production*, vol. 88, pp. 297–307, Feb. 2015, doi: 10.1016/j.jclepro.2014.04.036.

[13] I. T. Almalkawi, M. Guerrero Zapata, J. N. Al-Karaki, and J. Morillo-Pozo, "Wireless Multimedia Sensor Networks: Current Trends and Future Directions," *Sensors*, vol. 10, no. 7, Art. no. 7, Jul. 2010, doi: 10.3390/s100706662.

[14] S. R. Jino Ramson and D. J. Moni, "Applications of wireless sensor networks — A survey," in *2017 International Conference on Innovations in Electrical, Electronics, Instrumentation and Media Technology (ICEEIMT)*, Feb. 2017, pp. 325–329. doi: 10.1109/ICIEEIMT.2017.8116858.

[15] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras, "Applications of Wireless Sensor Networks: An Up-to-Date Survey," *Applied System Innovation*, vol. 3, no. 1, Art. no. 1, Mar. 2020, doi: 10.3390/asi3010014.

[16] C. C. Y. Poon, B. P. L. Lo, M. R. Yuce, A. Alomainy, and Y. Hao, "Body Sensor Networks: In the Era of Big Data and Beyond," *IEEE Reviews in Biomedical Engineering*, vol. 8, pp. 4–16, 2015, doi: 10.1109/RBME.2015.2427254.

[17] V.-T. Ninh, "Stress Detection in Lifelog Data for Improved Personalized Lifelog Retrieval System".

[18] A. Bruun and M. L. Stentoft, "Lifelogging in the Wild: Participant Experiences of Using Lifelogging as a Research Tool," in *Human-Computer Interaction – INTERACT 2019*, D. Lamas, F. Loizides, L. Nacke, H. Petrie, M. Winckler, and P. Zaphiris, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 431–451. doi: 10.1007/978-3-030-29387-1_24.

[19] Y. Cho *et al.*, "Platform design for lifelog-based smart lighting control," *Building and Environment*, vol. 185, p. 107267, Nov. 2020, doi: 10.1016/j.buildenv.2020.107267.

[20] S. Sathyanarayana, R. K. Satzoda, S. Sathyanarayana, and S. Thambipillai, "Vision-based patient monitoring: a comprehensive review of algorithms and technologies," *J Ambient Intell Human Comput*, vol. 9, no. 2, pp. 225–251, Apr. 2018, doi: 10.1007/s12652-015-0328-1.

[21] D. Tang, Y. Yoshihara, T. Takeda, J. Botzheim, and N. Kubota, "Informationally Structured Space for Life Log Monitoring in Elderly Care," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 2015, pp. 1421–1426. doi: 10.1109/SMC.2015.252.

[22] N. N. Malik, W. Alosaimi, M. I. Uddin, B. Alouffi, and H. Alyami, "Wireless Sensor Network Applications in Healthcare and Precision Agriculture," *Journal of Healthcare Engineering*, vol. 2020, p. e8836613, Nov. 2020, doi: 10.1155/2020/8836613.

[23] P. Rajak, A. Ganguly, S. Adhikary, and S. Bhattacharya, "Internet of Things and smart sensors in agriculture: Scopes and challenges," *Journal of Agriculture and Food Research*, vol. 14, p. 100776, Dec. 2023, doi: 10.1016/j.jafr.2023.100776.

[24] "ABX00042-ABX00045-ABX00046-datasheet.pdf." Accessed: Mar. 11, 2024. [Online]. Available: https://docs.arduino.cc/resources/datasheets/ABX00042-ABX00045-ABX00046-datasheet.pdf

[25] "Portenta Vision Shield | Arduino Documentation." Accessed: Mar. 11, 2024. [Online]. Available: https://docs.arduino.cc/hardware/portenta-vision-shield/

[26] "Arduino MKR ENV Shield rev2," Arduino Online Shop. Accessed: Jun. 10, 2023. [Online]. Available: https://store-usa.arduino.cc/products/arduino-mkr-env-shield-rev2

[27] "Gravity: PPG Heart Rate Monitor Sensor for Arduino (Analog/Digital)." Accessed: Nov. 13, 2024. [Online]. Available: https://www.dfrobot.com/product-1540.html

[28] "DS3231 RTC (Real Time Clock) Interfacing with Arduino to build DIY Digital Clock." Accessed: Aug. 19, 2023. [Online]. Available: https://circuitdigest.com/microcontroller-projects/interfacing-ds3231-rtc-with-arduino-and-diy-digital-clock

[29] "Getting Started with Arduino IDE 2 | Arduino Documentation." Accessed: Mar. 11, 2024. [Online]. Available: https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/

[30] G. Cicirelli, R. Marani, A. Petitti, A. Milella, and T. D'Orazio, "Ambient Assisted Living: A Review of Technologies, Methodologies and Future Perspectives for Healthy Aging of Population," *Sensors*, vol. 21, no. 10, Art. no. 10, Jan. 2021, doi: 10.3390/s21103549.

[31] G. Marques and R. Pitarma, "Promoting Health and Well-Being Using Wearable and Smartphone Technologies for Ambient Assisted Living Through Internet of Things," in *Big Data and Networks Technologies*, Y. Farhaoui, Ed., Cham: Springer

International Publishing, 2020, pp. 12–22. doi: 10.1007/978-3-030-23672-4_2.

[32] R. Maskeliūnas, R. Damaševičius, and S. Segal, "A Review of Internet of Things Technologies for Ambient Assisted Living Environments," *Future Internet*, vol. 11, no. 12, Art. no. 12, Dec. 2019, doi: 10.3390/fi11120259.

[33] G. Menculini *et al.*, "Insights into the Effect of Light Pollution on Mental Health: Focus on Affective Disorders—A Narrative Review," *Brain Sciences*, vol. 14, no. 8, Art. no. 8, Aug. 2024, doi: 10.3390/brainsci14080802.

[34] N. U. Huda, I. Ahmed, M. Adnan, M. Ali, and F. Naeem, "Experts and intelligent systems for smart homes' Transformation to Sustainable Smart Cities: A comprehensive review," *Expert Systems with Applications*, vol. 238, p. 122380, Mar. 2024, doi: 10.1016/j.eswa.2023.122380.

[35] K. Bhui *et al.*, "Air quality and mental health: evidence, challenges and future directions," *BJPsych Open*, vol. 9, no. 4, p. e120, Jul. 2023, doi: 10.1192/bjo.2023.507.

[36] G. Facchinetti, G. Petrucci, B. Albanesi, M. G. D. Marinis, and M. Piredda, "Can Smart Home Technologies Help Older Adults Manage Their Chronic Condition? A Systematic Literature Review," *International Journal of Environmental Research and Public Health*, vol. 20, no. 2, p. 1205, Jan. 2023, doi: 10.3390/ijerph20021205.

[37] J. Radua, M. De Prisco, V. Oliva, G. Fico, E. Vieta, and P. Fusar-Poli, "Impact of air pollution and climate change on mental health outcomes: an umbrella review of global evidence," *World Psychiatry*, vol. 23, pp. 244–256, May 2024, doi: 10.1002/wps.21219.

[38] A. A. Abba Ari *et al.*, "Data collection in IoT networks: Architecture, solutions, protocols and challenges," *IET Wireless Sensor Systems*, vol. 14, no. 4, pp. 85–110, Aug. 2024, doi: 10.1049/wss2.12080.

[39] M. Mohammed, M. N. Mahmud, M. F. Mohd Salleh, and A. Alnoor, "Wireless sensor network security: A recent review based on state-of-the-art works," *International Journal of Engineering Business Management*, vol. 15, Feb. 2023, doi: 10.1177/18479790231157220.

Journal of Soft Computing and Artificial Intelligence
Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access
Volume 05
Issue 02
December, 2024

Research Article

Hybrid Bee Colony Algorithm with Whale Algorithm

Radhwan Basim Thnoon [1,2]

[1]*Department of Mathematics, College of Computer Science and Mathematics, University of Mosul, Iraq*
[2]*Nineveh Education Directorate, Ministry of Education, Nineveh, Iraq*

ARTICLE INFO

Article history:
Received **June 24, 2024**
Revised **September 5, 2024**
Accepted **September 23, 2024**

Keywords:
Optimization
Bee-Colony Algorithm
Whale Optimization Algorithm
Hybrid Algorithm

ABSTRACT

In this paper, we introduce two hybrid meta-heuristic algorithms inspired by natural processes: Bee Colony Optimization (BCO) and Whale Optimization Algorithm (WOA). The BCO algorithm, first proposed by Karaboga in 2005, draws on the foraging behaviors of honeybees. It is known for its simplicity and effectiveness in solving various optimization problems. We will provide an overview of the BCO algorithm, including its principles and modifications within the context of swarm intelligence. This technique, which examines decentralized systems composed of numerous interacting elements, is particularly notable for its exploration capabilities. The Whale Optimization Algorithm, introduced by Mirjalili and Lewis in 2016, mimics the bubble-net hunting behavior of humpback whales. This algorithm employs swarm intelligence to avoid local optima and balance exploration and exploitation through a simulated fishing net approach. Its design helps achieve optimal solutions and effectively avoid local traps.We describe a hybridization of BCO and WOA into a new algorithm, named ABCWOA. This hybrid algorithm was tested across 16 optimization tasks with varying frequencies of (100, 200, 500, 1000). The results demonstrated that ABCWOA effectively reached optimal solutions, often outperforming traditional search algorithms by achieving lower minimum values (f_min) for most tasks.

1. Introduction

Several optimization issues dealing with different search areas of areas such as feature selection have been applied. Reduce dimensions. Mass transportation services. Recently, many Meta-Heuristic algorithms have been developed because of their advantages that help researchers propose a solution to complex computational problems. Such as Flock Amendment. inborn algorithms. Bat algorithm (BAT). Ant-Lion Algorithm (ALO). And other post intuitive algorithms because of their advantages of flexibility in dealing with different problems compared to traditional algorithms make optimization techniques in these algorithms popular among researchers [1]. Post-intuitive optimization algorithms have become more widely used in applications related to engineering because they are based on easy concepts that are plain to perform and do not demand tendency inclination and can also avoid falling into local solutions. Post-intuitive algorithms inspired by optimization problems are solved by simulating biological or materialistic events. They can be set into three major class: growth-founded methods. and materialistic based. And the list of the flock.[2]

In post intuitive algorithms, it relies on the random population principle in exploration and exploitation. Post-intuitive algorithms work on less computational Access time the optimal resolution, shun descent till solutions with faster convergence, and take out the mundial optimal resolution that makes it Adecuado for practical applications without modifications in

* Corresponding author
e-mail: rathwanbasim@gmail.com
DOI: 10.55195/jscai.1503982

the structure of the algorithm for a different solution that is beneficial or not useful in algorithm optimization problems.[3]

2. Research Problem

The research problem focused on finding the comprehensive optimal solution for unrestricted high-measurement optimization problems without falling into the trap of local solutions. and its characteristics to solve the problem of dropping into topical solutions.

3. Research Importance

The value of the discussion is in beneficent the showing of the algorithms by interbreeding them between the bee optimization algorithm and the Whale Optimization Algorithm and it was called (ABCWOA). It is a suggested method for solving unrestricted optimization problems with a high measurement, which is considered one of the difficult (NP-hard) problems, to help researchers and specialists to benefit from the proposed hybrid algorithm in solving this type of problem, as well as all the issues that fall within its framework.

4. Bee Colony Algorithm

4.1. Behavior of bees: Behavior Bee

Honeybees live in and maintain a densely populated colony. It is a uniquely explained about social organization. They are creatures of unexpected complexity capable of generating food. and explore food sources. And communicating through complex dances, the colony can be considered a unique swarm of social elements, the bees, who exchange information with them, leading the bees to form collective knowledge. A bee colony consists of approximately one queen, hundreds of drones and many thousands of workers. The miss lays false eggs and an alchemical plant. Her labor is to spread bleaches and onset a modern colony. She tampers conduct by sending (chemical messages) [4]. The Bee order is a gauge model of neat teamwork, coordination, interaction, division of labor, and continuous performance of tasks in the colony.[5]

Despite the large digit of diverse gregarious mealy bug host and the divergence in their types of behavior, individual insects can be described as able to carry out a diversity of synthesis tasks. The superior ideal is collecting and converting nectar, which is much more orderly. Each bee chooses to gauge the source of nectar by next a special system among them in distributing the tasks and duties required of them. The colony consists of [6] the queen and her responsibility are to lay eggs so that they form a new colony. The males are responsible for mating with the miss and this is their only turn in the hive. They are eliminated from the habitation over their downfall. The laborer bees are the feminine of the stand. They are the basic erection blocks of the stand that builds the melissa bee reed, cleans, maintains, protects and feed the talent and males. Aloof from this aspect of responsibilities, the prime act of the laborer bee is to seek for and converge opulent food. There are two forms of laborer bees, prospector bees and elemental feeder bees.[5]

Prospector bees take wing and food fountains randomly from flowers and nectar. They reversion to the hive after depleting their power and the edges of distance, and when they reversion to the hive, they portion the experiment of discovery from the remarkable datum with the nutrient element, the prospectors tell the foragers about the position of the opulent food roots that make up the orientation (nook) of the food root from the hive to the bask and the interval from the hive and it is done This is done through the use of a dance [5]. This dance changes with an increase in the distance to the food source, and different types of this behavior have been identified, including

- **Circular dance**
- **Waggle tail dance**

And each type of them can have degrees or differences that illustrate a certain behavior, as in Figure (1) the bee in the circular dance runs in a circular motion, where several circles are made by the bee with the making of bubbles when giving samples of nectar to the bees. Dance shows the location of a source within a certain distance. In the case of food close to the colony.[7] If the food is far from the colony, the circular dance is developed into the wagging dance or the wagging tail dance, which is in the form of the number 8 by imposing providing the beehive with information about the distance and direction of the food with running in a circle in one direction and then in a circular run in the other direction with shaking the body during a straight run . These dances are performed on a horizontal disc called the dance area, which is a specific place in the colony[7] .

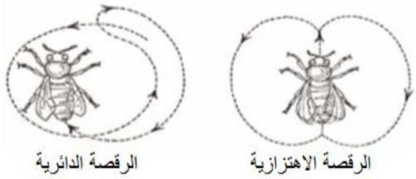


Figure 1 Illustration image of bee dances.

4.2. Bee Colony System

It is an inspiring exploratory method developed to find effective solutions to the difficulty of getting involved in optimization problems. The requisite motif backwards a bee colony system is to build a multi-component system (synthetic Bee habitation) that will look for perfect resolution for unlike improvements, and explore the principles that honey bees use while collecting nectar. A synthetic bee colony ordinarily depends on a few individuals. To find the best possible solutions in a given search area, artificial bees cooperate by exchanging information and food locations using“ waggling dances.[8]

4.3. Synthetic Bee habitation Algorithm

The ABC synthetic bee-colony algorithm was suggested by Karaboga in 2005 for true preference criteria. It is an Optimization algorithm that affects the feeding manner of a bee colony [7]. Each bee is a simple component of the ABC. In the formation of a bee colony with these simple components, it can display complex and coherent behavior and to establish an integrated system for its detection and exploration of flower nectar. Each habitation has three brackets of prospector bee, sighting bees, and workers each bees with an onus. Prospector bees must search and find new sources of food. The search is random about the surrounding environment [9]. The top bisection of the habitation includes the used Synthetic bees and the latter bisection implicates the observer bees. For each source of nourishmto ent yonder is a singular bee working on the other hand, the numeral of worker bees equals the numeral of nourishment provenance. Worker bees determine the food provenance by those memories. Worker bees part their acquaintance with the spectator bees in the hive and the spectator bees choose a single source of food. The worker bee that has been derelict from this source becomes a prospector that else initiates a random search for new nourishment sources.[10]

Each research cycle includes three phases: transferring the spectators (watching bees) and working bees to food sources, in addition to calculating the quantities of their nectar, as well as marking the prospector bees, and then randomly transferring them to imaginable nourishment provenances. The nourishment provenance appears a scientific resolution to an issue improve it. The compensation of nectar of the nourishment provenance agrees to the fineness of the food source. Viewer bees are placed on foods using a dancing disc. Every hive has prospectors who are scouts for the habitation, scouts don't have any leading when manhole for habitation they are only interested in feedback on any type of nourishment origin. As a result of this manner, prospectors have lower search costs as well as lower average food source quality.

Sometimes scout bees can accidentally discover rich food sources[10] .

The manner of the BCO bee habitation is stagy The ABC synthetic bee colony algorithm differs from BCO because in ABC we use but prospectors and foragers in equal proportions as the inhabitance priority. The prime proceedings of the ABC algorithm are:

- config
- Repetition
- Send worker bees nourishment origin
- Toss off watch bees towards food sources
- Toss off prospector bees to explore modern nourishment origin
- obtaining the wished-for (appropriate) status[9]

step 1:

Bisection of the bees in the ABC Algorithm includes worker bees and the other bisection includes no-worker bees. There is lone singular cohort of worker bees for either food origin.

Step 2:

Use the following relationship. That a modern return was formed solution for either trouble

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \qquad (1)$$

where

$i\ ,\ \ k\,\epsilon\{1.2\ldots.BN\}\ \&\ k \neq i$

$j\,\epsilon\{1.2\ldots.D\}$

$\varphi\epsilon[-1,1]$

The vicinity of the nourishment origin is performed by x_i, the nourishment origin v_i has to be a value of x_i, j an is an indiscriminate integer in the zone of $[1, D]$, D is the number of optimization coefficients, and K is a randomly selected index whose range is $[1, BN]$, BN is the issue of premier answers to the trouble, and unlike i , φ is a true random issue in the range $[-1,1]$.[9]

Step 3:

$$p_i = \frac{fit_i}{\sum_{n=1}^{sn} fit_n} \qquad (2)$$

The eventuality of receiving bees from each site is calculated using the following equation (minimization problem)

Where fit_i: is the fitness function and is the source of the fit, p_i is the eventuality of election by the observer bees, the numeral of bees is allocated pacting to the fit of any matter in this tread, and all bees may be assigned to a feeding place rule on the fitness function. next any origin

***Step 4**:*

The indiscriminate response will replace the no-optimized response, if the number of no-optimizing responses reaches a preset edge C_{max}. Moreover, the iteration finish events are investigated. At this scene the iterations will be terminated as long as the algorithm end events are specified, otherwise They will recur to another pace.[9]

5. Whale Optimization Algorithm (WOA)

The Whale Optimization Algorithm is a modern algorithm proposed by Mirjalil and Lewis (2016) that simulates the attitude of humpback whales in their discussion for nourishment and hunting. Whales are amazing world, and they are gazed the largest mammals in the cosmos, where a grown-up whale mastery magnifies up to 30 meters in length and 180 tons in metering. Whales are mostly predators, and they are animals that do not shut eye so that they have to respire from the flat of the peripheries. And that one of the largest baleen whales are humpback whales, where the size of a grown-up humpback whale is nearly the magnitude of a schoolmistress bus, and that their preferable victim is (Krill) and modicum flocks of fish. Figure (2) display these mammals[2] .

The generality enjoyable thing about humpback whales is their chasing style. This foraging demeanor is called bleb decoy nourishment. The humpback whale prefers to chase the aquatic herd of (Krill) or modicum fish close to the superficies, and it has been monitored that this for-aging behavior is well-done by creating distinct blebs over an area or path in the form of 9 as shown in the numbered figure 2[2](2) .

2

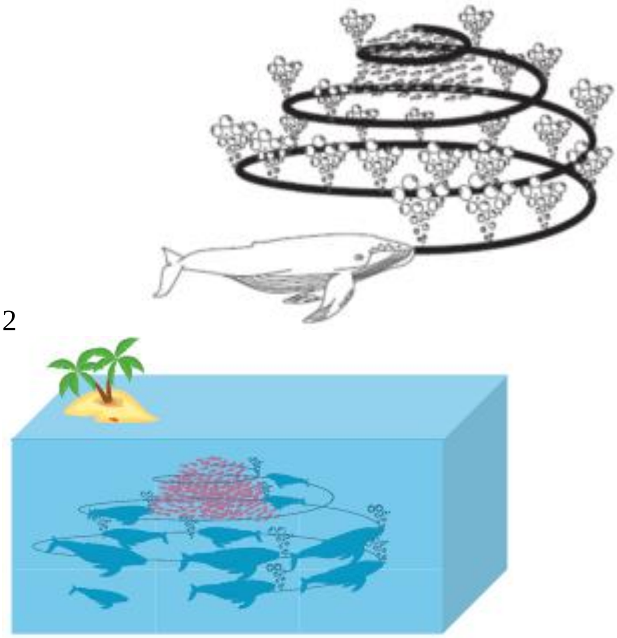


Figure 2 bleb decoy nourishment demeanor in humpback whales

The arithmetical example of the Whale-Optimization-Algorithm can be summarized in the following paragraphs:

5.1. Encircling prey

Humpback whales can meet the position of victim and surround it to hunt it, and this demeanor is mathematically expressed by the following poise:

Since:

$$\vec{D} = \left|\vec{C}.\vec{X}^*(t) - \vec{X}(t)\right| \qquad (3)$$

$$\vec{X}(t+1) = \vec{X}^*(t) - \vec{A}\vec{D} \qquad (4)$$

t denotes the current iteration, $\vec{X}^*$: is the position vector of the best solution obtained so far, X: is the position vector, and the symbol | | It refers to the absolute value, and the symbol (.) represents the product of multiplying an element by an element. In addition, A, C: refer to coefficient vectors and are calculated using the equations below 11]

$$\vec{A} = 2\vec{a}.\vec{r} - \vec{a} \qquad (5)$$

$$\vec{C} = 2\vec{r} \qquad (6)$$

As it $\vec{a}$ decreases linearly from 2 to 0 over the iterations (in both stages of exploration and exploitation) and that it is a random vector that takes values [0,1]. It is worth noting that X ⃗^* should be updated every iteration if there is a better solution [12].

5.2. Bubble-net attacking method

In tidy to arithmetically perform the demeanor of humpback whales in bubble decoy hunting, a hybrid was designed:

. Shrinking encircling mechanism

This behavior $\vec{a}$ is achieved by decreasing the value $\vec{a}$ in equation (5) noting that the fluctuation $\vec{A}$ is also decreasing by $\vec{a}$. And be calculated by $\vec{a}$ the following equation:

$$a = 2 - t\frac{2}{MaxIter} \qquad (7)$$

Where: t: indicates the current iteration and $MaxIter$: represents the largest number of iterations allowed.

. Spiral updating position

Figure 9 or the shape of the helix is simulated using the following equation:

$$\vec{X}(t+1) = \vec{D}.e^{bi}\cos(2\pi) + \vec{X}^*(t) \qquad (8)$$

Since:

$\vec{D}$:denotes the distance *i* from the whale to the prey (the best solution obtained so far), b is a constant for determining the shape of the logarithmic vortex, l is a random number in the interval [-1,1], and the sign (.) is the product of two components. Figure (3)

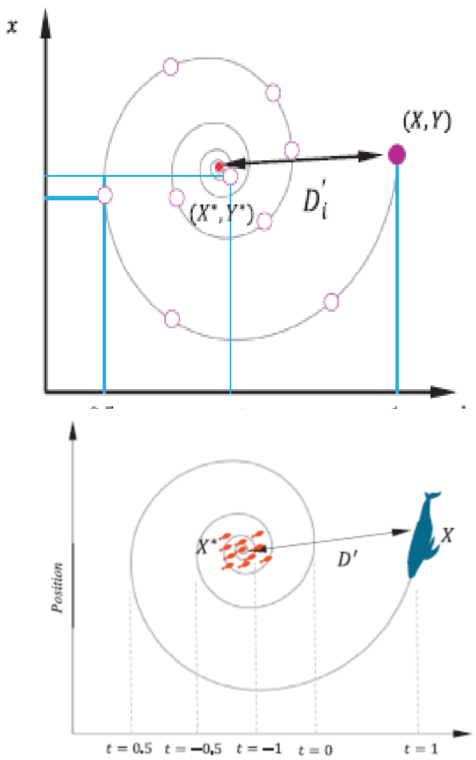


Figure 3 The spiral site

It has been observed that humpback whales bathe about the victim into a contraction ring and straight a cochleate-shaped track altogether. To paradigm this demeanor altogether, we suppose a 50% eventuality of choosing amidst either a contraction encircling technicality or a winding paradigm to update the whales' location midst optimization. The arithmetical paradigm is as follow:

$$(t+1) = \begin{cases} \vec{X}^*(t) - \vec{A}\vec{D} & if\ p < 0.5 \\ \vec{D}.e^{bi}\cos(2\pi) + \vec{X}^*(t) & if\ p \geq 0.5 \end{cases} \qquad (9)$$

Where: p is a random number in the interval.[0,1]

5.3. Search for prey

Humpback whales, as well as bubble decoys, search for prey randomly according to their respective locations, so we randomly update the finder location in the exploration phase instead of the best finder found so far. So we use a vector $\vec{A}$ with indiscriminate merits maximal than 1 or not so much than -1 to impact the cast around the element to walk from here from the hint whale. This mechanism and $|\vec{A}| > 1$ emphasizes the discovery process and allows the Whale Optimization Algorithm (WOA) to carry out a comprehensive scout out. The arithmetical paradigm is as follows up:[2]

$$\vec{D} = |\vec{C}.\vec{X}^*(t) - \vec{X}(t)| \qquad (10)$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A}\vec{D} \qquad (11)$$

Since

$\overrightarrow{X_{rand}}$ is the random vector (the random whale) that was chosen from the current population.

6. Hybrid Algorithm:

Swarm intelligence is a section of synthetic cleverness that projects the mass demeanor and emerging characteristics of multiplex, Self-organizing and decentralized frameworks with a friendly frame. This system consists of modest interactive factors marshaled in small communities called (flocks), even though each element has a specific area to work and there is no centric observation, and the accumulated demeanor is one of the characteristics of imposed in the swarm. They can respond to ecological variations and rebate making abilities.

In this section, I proposed a new hybrid algorithm by linking post-intuitive ideas inspired by nature. Generally, the synthetic Bee settlement (ABC)

Algorithm was hybridized with Whale Optimization (WOA),

which uses swarm intelligence to get it the overall optimal resolution to optimization troubles. The new proposed algorithm was named (ABCWOA).

7.Practical side

The qualification of the proposition new algorithm (ABCWOA) was examined by using it to solve (16) high-measurement numerical optimization problems.

The new IWOMFO algorithm was compared with the synthetic Bee settlement algorithm (ABC) and the Whale Optimization Algorithm (WOA). The table shows 1) specifics of the trial functions, as well as the special domain, the minimum value (fmin) for each function, the slash and celestial limits for each function, the number of iterations (100, 200, 500, and 1000) reestablishment were used, and the remoteness that set forth the numeral of changeable in the styling. Elements. The proposition new algorithm (ABCWOA) is an amalgamation of two algorithms synthetic Bee settlement (ABC) and Whale Optimization Algorithm (WOA). Tables numbered from (2) to (4) show the values of the outputs.

Table 1 Details of the parameters and the lower and upper limits

Function	Dim	Range	*Fmin*
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100,100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	30	$[-10,10]$	0
$F_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_i)^2$	30	$[-100,100]$	0
$F_4(x) = \max_i\{\lvert x_i\rvert . 1 \le i \le n\}$	30	$[-100,100]$	0
$F_5(x) = \sum (100 * (x_{i+1}:\text{dim} \quad) - (x_i:\text{dim} \quad)^2 + ((x_{i+1}:dim-1)-1)^2$	10	$[-30,30]$	
$F_6(x) = \sum_{i=1}^{n} ([x_i \quad + 05])^2$	30	$[-100,100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	30	$[-1.28,1.28]$	0
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i\rvert})$	30	$[-500,500]$	-418.9829
$F_9(x) = \sum_{i=1}^{n} \lvert x_i^2 - 10\cos(2\pi x_i) + 10\rvert$	30	$[-5.12,5.12]$	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	$[-32,32]$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-600,600]$	0
$F_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_i) + \sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1}) + (y_n-1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	30	$[-50,50]$	0
$y_i = 1 + \frac{x_i+1}{4} u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i+a)^m & x_i < a \end{cases}$			
$F_{13}(x) = 1.0\{\sin^2(3\pi x_i) + \sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(\pi x_{i+1}) + (x_n-1)^2[1+\sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)\}$	30	$[-50,50]$	0
$F_{14}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1 - (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^4$	4	$[-5,5]$	0.00030
$F_{15}(x) = 4x_i^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5,5]$	-1.0316
$F_{16}(x) = \left[1 + (x_1 \; + x_2 \; + 1)^2(19 - 14x_1 \; + 3x_1^2 - 14x_2 \; + 6x_1x_2 + 3x_2^2)\right] * \left[30 + (2x_1 \; - 3x_2)^2(18 - 32x_1 \; + 12x_1^2 + 48x_2 \; - 36x_1x_2 + 2x_2^2)\right]$	2	$[-5,5]$	-1.0316

Table 2 Using frequency 100 and number of items 5

Function	ABC	WOA	ABCWOA
F1	1.0444e-10	2.0246e-41	0
F2	9.6791e-07	1.6006e-25	0
F3	1.7878e-07	70121.1481	1.106e-272
F4	1.5448e-05	79.9442	6.8175e-28
F5	0.0040749	28.7177	0.025852
F6	1.5692e-10	1.4552	7.2226e-08
F7	0.0052746	0.0019287	0.60139
F8	-2230.9123	-8713.2418	-2094.6025
F9	6.9647	0	0
F10	5.1781e-06	4.4409e-15	4.4409e-15
F11	0.0098573	0	0
F12	3.9099e-12	0.046243	0.00024966
F13	1.9942e-11	0.83173	0.00012812
F14	0.00069064	0.00033453	0.00031245
F15	-1.0316	-1.0316	-1.0316
F16	3	3	3

Table 3 Using frequency 250 and number of items 5

Function	ABC	WOA	ABCWOA
F1	3.485e-06	1.0413e-15	0
F2	0.00023305	8.8026e-12	0
F3	0.0043285	86184.8331	5.9759e-152
F4	0.0017774	1.0603	1.5683e-09
F5	0.12367	28.7605	0.076329
F6	6.2318e-06	2.2564	3.4758e-07
F7	0.01054	0.0039246	0.15095
F8	-2094.9144	-7918.9488	-2094.6058
F9	6.9647	0	0
F10	0.00088075	1.3146e-07	7.9936e-15
F11	0.0074123	1.2879e-14	0
F12	4.5156e-09	0.25834	5.7205e-05
F13	2.2886e-07	0.76708	0.00019614
F14	0.00031911	0.011256	0.00075824
F15	-1.0316	-1.0316	-1.0316
F16	3	3.0017	3

Table 3 Using frequency 500 and number of items 5

Function	ABC	WOA	ABCWOA
F1	2.2539e-24	2.4774e-105	0
F2	1.7809e-13	4.0432e-71	0
F3	8.4283e-20	42743.963	2.4703e-323
F4	1.6095e-12	46.0478	5.3762e-71
F5	0.94103	28.7527	0.00043173
F6	2.9983e-24	1.3531	3.1469e-09
F7	0.0033792	0.00015082	0.8238
F8	-1897.5128	-12334.9275	-2094.8756
F9	6.9647	0	0
F10	1.3154e-12	4.4409e-15	8.8818e-16
F11	0.0098647	0	0
F12	3.6252e-25	0.086897	4.5053e-05
F13	2.573e-25	0.59082	7.8302e-06
F14	0.00052025	2.9821	0.00032294
F15	-1.0316	-1.0316	-1.0316
F16	3	3	3

8.Discussion of Numerical Results

The results in the previous table show the success of the hybrid algorithm (ABCWOA) in feedback on the optimal resolution for (16) of the standard test functions with a high measurement compared with the synthetic Bee settlement algorithm again and again with the Whale Optimization Algorithm WOA, and this confirms the success of the hybridization process. The proposition algorithm ABCWOA gave better results from the algorithm ABC and WOA, we note that the functions (F11, F9, F2, F1) have given an ideal solution that is better than the algorithm itself, i. The algorithm of whale optimization, but it improves in the hybrid algorithm. As for the functions (F11, F8, F7), we notice an improvement in the Whale Optimization Algorithm, and it gets worse in the synthetic bee colony algorithm. As for the functions (F15, F14, F10), they may be bad with constancy in the value of convergence when crossing.

Table 4 Using frequency 1000 and number of items 5

Function	ABC	WOA	ABCWOA
F1	9.536e-47	1.0397e-208	0
F2	1.8945e-24	3.0999e-136	0
F3	1.5979e-18	22776.1519	2.9644e-323
F4	2.1646e-23	79.0986	1.3765e-224
F5	0.11947	28.7487	0.00063479
F6	0	0.62392	1.9829e-08
F7	0.01665	1.4017e-05	0.028716
F8	-2252.8289	-12568.7956	-2094.9129
F9	1.9899	0	0
F10	4.4409e-15	4.4409e-15	4.4409e-15
F11	0.0098573	0	0
F12	9.4233e-32	0.0198082	6.5582e-08
F13	1.3498e-32	0.67866	8.543 e-15
F14	0.00071787	0.00035035	0.00032294
F15	-1.0316	-1.0316	-1.0316
F16	3	3	3

9. Conclusions and recommendations

First: Conclusions

The hybridization of the meta-heuristic evolutionary algorithm with the heuristic algorithm enhanced the performance of the combined approach by accelerating convergence and improving solution quality. This integration increased both the exploration and search capabilities of the algorithm. Numerical results demonstrated the effectiveness of the hybrid algorithm in addressing various optimization problems. Comparative analysis between the Bee Colony Optimization (BCO) and Whale Optimization Algorithm (WOA) revealed that the hybrid algorithm consistently achieved superior results. The comprehensive optimal solutions were attained for most test functions, as indicated by the results with a difference with other algorithms (F16).

Second: Recommendations

In light of the conclusions reached by this research, the following recommendations can be presented:

1. Attempt to hybridize the ant colony algorithm with other algorithms that depend on the swarm system and improve it.
2. Attempt to hybridize the bee colony algorithm with classical optimization algorithms and compare the results of this hybrid algorithm with the hybrid algorithm with the swarm.
3. Attempt to hybridize the whale optimization algorithm with classical optimization algorithms and compare the results of this hybrid algorithm with the hybrid algorithm with the swarm.
4. Apply the proposed hybrid algorithm to one of the operations research problems of the (NP-hard) type

10.Reference

[1]. Hussien, A.G., et al., *New binary Whale Optimization Algorithm for discrete optimization problems.* Engineering Optimization, 2020. **52**(6): p. 945-959

[2]. Mirjalili, S. and A. Lewis, *The Whale Optimization Algorithm.* Advances in engineering software, 2021 .6 :95p. 51-67.2013.

[3] Trivedi, I.N., et al., *Novel adaptive Whale Optimization Algorithm for global optimization.* Indian Journal of Science and Technology, 2016. **9**(38): p. 319-326.

[4] Abraham, A., R.K. Jatoth, and A. Rajasekhar, *Hybrid differential artificial bee colony algorithm.* Journal of computational and theoretical Nanoscience, 2012. **9**(2): p. 249-257.

[5] Kaur, A. and S. Goyal, *A survey on the applications of bee colony optimization techniques.* International Journal on Computer Science and Engineering :(8)3 .2011 ,p. 3037.

[6] Teodorović, D., *Bee colony optimization (BCO)*, in *Innovations in swarm intelligence*. 2009, Springer. p. 39-60.

[7] Khaleel, S.I., *Selection and Prioritization of Test Cases by using Bees Colony.* AL-Rafidain Journal of Computer Sciences and Mathematics, 2014. **11**(1): p. 179-201.

[8] Davidović, T., D. Teodorović, and M. Šelmić, *Bee Colony Optimization-part I: the algorithm overview*. Yugoslav Journal of Operations Research, 2015. **25**(1): p. 33-56.

[9] Ghaleini, E.N., et al., *A combination of artificial bee colony and neural network for approximating the safety factor of retaining walls.* Engineering with Computers, 2019. **35**(2): p. 647-658.

[10] Al-Safi, A.H.S., Z.I.R. Hani, and M.M.A. Zahra, *Using A Hybrid Algorithm and Feature Selection for Network Anomaly Intrusion Detection.* Journal of Mechanical Engineering Research and Developments, 2021. **44**(4): p. 253-262.

[11] Mohammed, H.M., S.U. Umar, and T.A. Rashid, *A systematic and meta-analysis survey of Whale Optimization Algorithm.* Computational intelligence and neuroscience, 2019. **2019**.

[12] Kaur, G. and S. Arora, *Chaotic Whale Optimization Algorithm.* Journal of Computational Design and Engineering, 2018. **5**(3): p. 275-284.

Journal of Soft Computing and Artificial Intelligence

Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access

Volume 05
Issue 02

December, 2024

Research Article

Artificial Algae Algorithm for Clustering of Benchmark Datasets

Sahar Rashedi[1], Muhammed Eshaq Rashedi[2], Murat Karakoyun[3]

[1,3] *Department of Computer Engineering, Necmettin Erbakan University, Konya, Turkey*
[2] *Department of Mathematics, Faculty of Science and Letter, Kafkas University, Kars, Turkey*

ARTICLE INFO

Article history:
Received October 2, 2024
Revised October 21, 2024
Accepted October 31, 2024

Keywords:
Artificial Algae Algorithm
Clustering
Medical Datasets
Optimization

ABSTRACT

The best solution found for a problem under specific circumstances is called optimization. Algorithms for optimization can make the best use of the information at their disposal. Numerous optimization algorithms have been created thus far by researchers, and most of these algorithms are based on the characteristics of naturally occurring biological organisms. Optimization algorithms have proven to be highly effective in numerous fields, including finance, engineering, and medical. Apart from these applications, they have also been employed in data mining techniques including clustering and classification. In many different domains, the clustering method is widely applied. Finding the optimum cluster centers is the most crucial step in the clustering process. In this study, the Artificial Algae Algorithm (AAA) is used to perform the clustering procedure by using 12 datasets that were taken from the UCI Machine Learning Repository. For every dataset, the squared distance values between the cluster centers and the data were computed in order to assess the effectiveness of AAA. The study evaluated AAA's performance against that of the ALO, DEA, MFO, PSO, TSA, and WOA algorithms. According to the experimental results, AAA took the first place by obtaining the best average values (0.72 for f1-score, 0.75 for sensitivity and 0.88 for specificity) in all three metrics, clearly demonstrating its success in the clustering problem.

1. Introduction

One use of data mining is the clustering issue, which involves organizing the items in a dataset based on similarities (or differences). Even in cases when the group to which the data belong is unknown, clustering algorithms aid in breaking the data down into subsets based on shared characteristics [1], [2]. Clustering is mostly used to group things based on shared characteristics. When grouping data into clusters, the goal is to make sure that members of the same group are comparable to one another, while members of different groups are placed in different groups. The goal is for individuals within a cluster to be close to one another, whereas individuals outside of a cluster are farther apart [3]. Clustering is performed to the population when the grouping of the variables in the dataset is unclear. This population's n data samples are examined across p variables. People that share similar traits are grouped together and divided into clusters throughout the clustering process. The process of clustering makes it possible to aggregate observational data with little loss [4]. When performing cluster analysis, the first step is to select a similarity or distance criterion. Then, it must be decided which clustering technique will be used. The type of method for clustering which is utilized for the chosen approach is chosen in the following phase, and the number of clusters to be employed in interpreting the cluster results is decided in the last

* Corresponding author
e-mail: mkarakoyun@erbakan.edu.tr
DOI: 10.55195/jscai.1560068

step [5]. The Euclidean distance was used in this work to determine the separation between each data instance and the cluster center to which it belonged [6]. Numerous methods exist in the literature to address the clustering problem, as the text mentions. That means there are two main types of clustering algorithms: hierarchical and non-hierarchical. The number of clusters does not have to be predetermined when using hierarchical clustering techniques. After the clustering procedure is finished, the number of clusters is established using these methods. On the other hand, the number of clusters needed to complete the clustering process is necessary for non-hierarchical clustering algorithms. In conclusion, the division of n data instances into k clusters is the aim of non-hierarchical clustering. According to Boushaki, S. I., Kamel, N., and Bendjeghaba, the time complexity of hierarchical clustering methods is n^2, whereas non-hierarchical clustering methods have n complexity [7].

Optimization is the process of producing the most efficient result with limited resources under expected conditions. Due to the complexity of the problems, it is difficult to estimate all possible combinations for this generated result. Mathematical model approaches simplify the problems to solve these problems. They also make assumptions about the results to reduce the search space instead of all possible combinations. There can be a significant difference between the solution of this simplified and result-limited problem and the solution of the real problem. Using intuitive techniques, this drawback of mathematical models is removed. Optimization algorithms have proven to be highly effective in numerous fields, including finance, engineering, and medical. They have been utilized in parameter updates for algorithms like data mining's clustering and classification in addition to these other applications. In several domains, clustering is a commonly utilized technique. Finding the optimal cluster centers for the clustered data is the most crucial step in the clustering process. Intuitive methods provide higher quality solutions by searching in detail instead of limiting the solution space [8]. Intuitive methods do not always guarantee the best case. However, they are called the approximate calculation approach and the aim here is to find acceptable appropriate solutions.

In this study, the Artificial Algae Algorithm (AAA) has been used to solve clustering problems. In this method, data samples in un-clustered datasets are clustered based on cluster centers obtained through global search. In global search, various probabilities in the search space are examined to minimize the error between data sample clusters. The AAA algorithm has been applied to datasets from the UCI repository, including aggregation, banknote, blobs, ecoli, glass, iris, iris2d, ionosphere, seeds, vertebral2, vertebral3 and wine. The clustering performance of the AAA algorithm compared to the ALO, DEA, MFO, PSO, TSA and WOA algorithms is investigated.

Since the importance of the clustering, there are numerous studies in literature that handled this problem. In their study, Karami and Zapata utilized Particle Swarm Optimization (PSO) and K-Means algorithms to counter attacks in content-based networks. In this approach, input attacks are classified into clusters, and preventive actions are taken based on the results obtained. The authors noted that using the K-Means algorithm alone does not sufficiently optimize cluster centers, so they employed a two-stage approach. In the first stage, called "training stage," cluster centers are obtained using a combination of PSO and K-Means algorithms. In the second stage, called "detection stage," a new fuzzy method is used to identify anomalies in the data. The results show that this method outperforms other clustering algorithms in achieving optimal cluster centers and higher-quality detection rates [9]. In contrast to conventional approaches, Liu and Ban have introduced a novel way to solve the clustering problem that does not need predetermining the number of clusters. The foundation of this approach is the creation of a dynamic, two-dimensional topological graph that shows the connections between the data points in every group. The efficacy and success of the Liu and Ban approach are demonstrated by the experimental results [10]. Rahman and Islam have presented a novel approach to data clustering that combines the K-Means technique with the GA. Finding the ideal number of clusters in the clustering problem is the goal of this approach. The outcomes demonstrate how well our method finds superior cluster centers. The authors showed that, overall, their suggested strategy performs better than five other compared methods after testing it on 20 distinct datasets [11].

Tzortzis and Likas introduced the MinMax K-Means algorithm to overcome the local optimization problem in the classic K-Means algorithm. In K-Means, due to the random selection of initial centroids, the algorithm may get stuck in a local optimum and fail to reach the best possible solution. The MinMax K-Means algorithm addresses this issue by assigning weights to clusters based on variance values, and these weight values are optimized according to the desired objective. Experimental results show that the MinMax K-Means algorithm outperforms the classic K-Means algorithm in terms of accuracy and efficiency [12]. In their study, Maulik and Bandyopadhyay employed GA for data clustering. They used two separate datasets—one artificial and the other real—to test the GA-based clustering technique. The findings show that, on average, the GA-based clustering approach outperformed the K-Means algorithm on the datasets [13]. Van der Merwe and Engelbrecht created a new clustering technique using the PSO algorithm. They compared the PSO-based clustering method's performance with the K-Means approach after evaluating it on six distinct datasets. The outcomes show that in terms of accuracy and efficiency, the PSO-based clustering method performs better than the K-Means approach [14]. The K-Means Algorithm was presented by Shelokar and associates in 2004 as a solution to the clustering problem. Tests have been conducted on both synthetic and actual datasets using this approach. Furthermore, the K-Means algorithm's performance has been contrasted with well-known optimization techniques like GA, Tabu Search, and Simulated Annealing. The outcomes of the experiments indicate that the KA algorithm performs extremely well in data clustering [15]. The PSO technique was used by Omran and his colleagues to update the cluster centers in the K-Means algorithm. Test datasets for image segmentation have been used to evaluate this PSO-based K-Means technique. Furthermore, the PSO-based K-Means algorithm's performance has been contrasted with that of the PSO and K-Means algorithms alone. The PSO-based K-Means algorithm is quite effective at segmenting images, according to experimental findings [16]. Zhang et al. solved the clustering problem by applying the Bee Colony Optimization (BCO) algorithm, which was inspired by the behavior of honeybees. They contrasted the widely used optimization techniques GA, Simulated Annealing, Tabu Search, and PSO with the BCO-based clustering approach. Three datasets—the iris, wine, and thyroid—that are frequently utilized for clustering were used in this comparison from the UCI repository. The experimental findings demonstrate that, across all datasets, the BCO-based clustering algorithm performs better than alternative optimization-based clustering methods [17]. Mat and associates created an innovative and effective clustering method that imitates the hunting behavior of whales. Ten medical datasets from the UCI Machine Learning repository were clustered using the newly developed WOA-LF technique. The original WOA clustering method, fuzzy c-means, k-means, and k-medoids were compared with WOA-LF's clustering performance. According to the application results, WOA-LF performs better in clustering tasks overall and may be utilized as a substitute method [18].

2. Problem Definition

The technique of separating data into distinct groups (clusters) according to their commonalities is known as clustering. These collections of data are referred to as "clusters," and each cluster's data is more comparable to its own than it is to that of other clusters. One popular data mining method for gleaning useful information from massive volumes of data is clustering. Numerous industries, including marketing, bioinformatics, health, and pattern recognition, use clustering. One effective method for identifying structure in data is clustering. Clustering is a helpful method for data analysis in many domains, despite its limitations [19]. The clustering approaches can be categorized into two main topics.

2.1. Hierarchical Clustering

Hierarchical clustering methods are used to sequentially determine clusters by bringing units together at different stages and to determine which distance (or similarity) level determines which elements will be members of the clusters. Hierarchical clustering can be examined in two groups: agglomerative hierarchical clustering and divisive hierarchical clustering. Agglomerative hierarchical clustering considers each observation in the data as a cluster. The merging operations are continued until a single cluster is obtained. Divisive hierarchical clustering assumes that all units form a cluster at the beginning and gradually separates the units into clusters. In hierarchical clustering techniques, clusters are merged sequentially and once a group is merged with another, it is not separated again in subsequent steps. These techniques create a hierarchical structure for the variables under consideration. The number of clusters in hierarchical clustering techniques is decided visually [20].

2.2. Non-Hierarchical Clustering

It can be applied if the researcher has determined the number of clusters that will be significant or if the number of clusters is known in advance. The division of units into clusters in this clustering technique can be done at random. The units are assigned to their respective clusters using the clustering criterion once the number of clusters into which they can be divided has been determined. K-means clustering, k-medoids clustering and fuzzy c-means clustering are some examples of non-hierarchical clustering techniques [2], [20].

In this study, artificial algae algorithm and some other metaheuristic algorithms were applied for non-hierarchical clustering.

3. Artificial Algae Algorithm (AAA)

AAA is a nature-inspired optimization method that draws inspiration from artificial algal behavior. An artificial alga performs photosynthesis by moving in a helical pattern toward a light source, just like a genuine algae does. It has the ability to shift the dominant species, adapt to its surroundings, and procreate through mitosis. An artificial algal colony represents every solution in the issue space. Each algae colony has an equal amount of algae cells as the problem dimension. The optimum is reached when an algal colony finds the perfect solution. Three primary components comprise the artificial algae algorithm: helical movement, adaptability, and evolutionary process [21].

3.1. Helical Movement

Artificial algae cells move helically towards the light. The energy of each helical movement determines whether the colony will change its position in space. At the beginning of each cycle, energy is calculated in proportion to the colony size and this energy represents the quality of the solution. The movement of algae in one dimension is shown in Equation 1 and the movements in the other two dimensions are shown in Equations 2 and 3 [22].

$$x_{im}^{t+1} = x_{im}^{r} + \left(x_{jm}^{t} - x_{im}^{t}\right)\left(sf - \omega_i\right) \tag{1}$$

$$x_{ik}^{t+1} = x_{ik}^{t} + \left(x_{jk}^{t} - x_{ik}^{t}\right)\left(sf - \omega_i\right)\cos\alpha \tag{2}$$

$$x_{il}^{t+1} = x_{il}^{t} + \left(x_{jl}^{t} - x_{il}^{t}\right)\left(sf - \omega_i\right)\sin\beta \tag{3}$$

Here, m, k, and l are random numbers selected from the range [1, d]. xi, yi, and zi represent the x, y, and z coordinates of the ith algae colony, respectively. j is the index of a neighboring algae colony obtained by tournament selection. p is a real number selected from the range [-1,1]. α and β are randomly selected angles from the range [0, 2π]. sf is the shear force caused by viscous movement. $\mathcal{A}i$ is the frictional surface area of the ith algae colony, which is proportional to its size. The frictional surface is calculated as the surface area of the hemisphere surrounding the algae colony due to its spherical shape. The frictional surface is given by Equation 4.

$$\omega_i = 2\pi r_i^2 \tag{4}$$

$$r_i = \sqrt[3]{\frac{3s_i}{4\pi}} \tag{5}$$

where r_i is the radius of the hemisphere of the ith algae colony and S_i is its volume.

3.2. Adaptation

Adaptation is the process by which an algae colony that is not growing sufficiently tries to resemble the largest algae colony in the vicinity. The hunger level determined by the helical movement is used. The hunger level does not change in the colony that goes to a better solution, while the hunger level of the colony that worsens increases. After each helical movement, the colony with the highest hunger value undergoes adaptation (Equation 6 and 7). Whether or not adaptation occurs is determined by the Adaptation parameter (Ap). Ap is a fixed value in the range [0,1] and is compared to a random number in this range. If the number is less than the Ap parameter, the adaptation process is carried out [22].

$$starving^{t} = \text{argmax}\{starvation(x_i)\} \tag{6}$$

$$starving^{t+1} = starving^{t} + (biggest^{t} - starving^{t}) \times rand \tag{7}$$

Here, $starvation(x_i)$ represents the hunger level of the ith algae colony, $starvingt$ is the algae colony with the highest hunger value at time t, $biggestt$ is the largest algae colony at time t, and rand is a real number randomly generated from the range [0,1].

3.3. Evolutionary Process

Artificial algae cells grow, develop, and divide into two artificial algae cells when they receive sufficient light. Algae cells that do not receive enough light die after a while. The evolutionary process is the stage where an algae cell from the largest algae colony (which has found solutions with better fitness function values than other colonies) is copied to replace each dead cell of the smallest algae colony (which has found solutions with worse fitness function values than other colonies) during the search process. This process is carried out as in Equations 8-10.

$$biggest^t = \arg\max size(x_i^t),\ i = 1,2,\ldots N \tag{8}$$

$$smallest^t = \arg\max size(x_i^t),\ i = 1,2,\ldots N \tag{9}$$

$$smallest_m^{t+1} = biggest_m^{t+1},\ m = 1,2,\ldots D \tag{10}$$

Here, $smallest$ represents the smallest algae colony, $biggest$ represents the largest algae colony, and D represents the problem dimension.

4. Experimental Environment

In this section, the datasets and the comparison metrics are detailed presented.

4.1. Datasets

In this study, the performance of AAA was evaluated on biomedical datasets obtained from UCI. The characteristics of the aggregation, banknote, blobs, ecoli, glass, iris, iris2d, ionosphere, seeds, vertebral2, vertebral3 and wine datasets selected from the UCI datasets are given in Table 1.

Table 1 Datasets used in the study

Dataset	Samples	Attributes	Classes
Aggregation	788	2	7
Banknote	1372	4	2
Blobs	1500	4	4
Ecoli	336	7	8
Glass	214	9	6
Iris	150	4	3
Iris2D	150	2	3
Ionosphere	200	11	6
Seeds	210	7	7
Vertebral2	310	6	2
Vertebral3	310	18	2
Wine	178	13	3

A larger view may be obtained by merging many data pieces, a process known as **aggregation**. Many industries, including research, marketing, finance, and healthcare, can benefit from the usage of aggregation datasets. The Aggregation dataset has 788 data instances overall and includes 2 numerical characteristics and 7 classifications.

The **Banknote** dataset contains images of real banknotes from various currencies. Such datasets are commonly used in the fields of artificial intelligence and image processing for tasks like counterfeit banknote detection, banknote classification, and money counting. This dataset has 4 numerical features and 2 classes, and the Banknote dataset consists of a total of 1372 data instances.

The **Blobs** dataset is a type of dataset that represents clusters of multiple points in a two-dimensional space. These points are often referred to as "blobs" and typically have a circular or elliptical shape. Blobs datasets are commonly used in fields like image processing, computer vision, and machine learning. The Blobs dataset has 4 numerical features and 4 classes, and the Blobs dataset consists of a total of 1500 data instances.

The **Ecoli** dataset is a type of dataset used by researchers who study the characteristics and behavior of the Escherichia coli bacterium. These datasets are commonly used in fields like microbiology, biology, computer science, bioinformatics, and medicine. The E. coli dataset has 7 numerical features and 8 classes, and the E. coli dataset consists of a total of 336 data instances.

The **Glass** dataset is a dataset used to predict glass quality. This dataset contains data about the chemical composition and properties of glass relevant to the glass industry. It is commonly used to train and test machine learning algorithms. The Glass dataset has 9 numerical features and 6 classes, and the Glass dataset consists of a total of 214 data instances.

The **Iris** dataset is a dataset published in 1936 by British statistician and biologist Ronald Fisher, and is considered a classic example of multivariate data analysis. This dataset is commonly used as a test dataset to evaluate and compare the performance of classification algorithms. In particular, it is an ideal dataset for training and testing supervised learning algorithms because it has labeled data (each instance indicates which flower species it belongs to). This dataset has 4 features and 3 class information, and the values of the features are taken from the width and length of the petals of the iris flower. This dataset consists of 150 examples belonging to three different species of Iris flowers. These examples are equally divided into 3 classes. There are 50 examples in the first class, Iris Setosa, 50 examples in the second class, Iris Versicolour, and finally 50 examples in the Iris Virginica class. The classification of the iris flower is done with these data examples.

The **Iris2D** dataset is a derivative of the Iris dataset and is commonly used for visualization and training of machine learning algorithms. This dataset contains the first two features of the "Iris" dataset (sepal length and sepal width) and is used to understand the performance of classification or clustering algorithms on these features. The "Iris2d" dataset contains the two-dimensional features of each flower instance, such as sepal length and sepal width. This makes it easy to visualize the data on a two-dimensional plane and see how the flower species are grouped based on these two features. This dataset is particularly common for data visualization and evaluating the performance of classification algorithms. For example, it can be used to visualize the outputs of different classification or clustering algorithms to see if the data points are correctly grouped or classified. The Iris2d dataset has 2 numerical features and 3 classes, and the Iris2d dataset consists of a total of 150 data instances.

The **Ionosphere** dataset is particularly used to evaluate the performance of classification algorithms. For example, a machine learning model can try to predict the state of degradation of an ionospheric radar signal using the features in the dataset. The Ionosphere dataset is widely used for training, research, and testing purposes in the fields of machine learning and data mining. The Ionosphere dataset has 11 numerical features and 6 classes, and the Ionosphere dataset consists of a total of 200 data instances.

The **Seeds** dataset is a dataset found in the UCI Machine Learning Repository. This dataset was obtained for use in agricultural research and contains seeds of three different wheat types (Kama, Rosa, and Canadian) in total. This dataset can be used to evaluate the performance of classification algorithms, to distinguish between wheat types, or for use in agricultural research. The Seeds dataset is particularly commonly used for training and evaluating machine learning classification algorithms. The Seeds dataset has 7 numerical features and 7 classes, and the Seeds dataset consists of a total of 210 data instances.

The **Vertebral2** dataset is a dataset containing information about vertebrae. This dataset can be used to develop artificial intelligence and machine learning models that can be used in the diagnosis and treatment of spinal diseases. The Vertebral2 dataset has 6 numerical features and 2 classes, and the Vertebral2 dataset consists of a total of 310 data instances.

The **Vertebral3** dataset is an important tool in spinal disease research. It can be used to improve the diagnosis, treatment, and prevention of spinal diseases. The Vertebral3 dataset has 18 numerical features and 2 classes, and the Vertebral3 dataset consists of a total of 310 data instances.

The **Wine** dataset is a popular dataset in the fields of machine learning and data science. It can be used for various tasks such as classifying wine types, predicting wine quality, and predicting wine price. It is particularly commonly used in classification and regression problems. The Wine dataset has 13 numerical features and 3 classes, and the Wine dataset consists of a total of 178 data instances.

4.2. Comparison Metrics

In this study, three metrics used to compare the performances of the algorithms. On the other hand, sum squared error value was used as fitness function.

4.2.1. Sum Squared Error (SSE)

SSE (Sum Squared Error) is a metric used in fields such as statistics and machine learning. It is commonly used to evaluate the performance of regression models. SSE represents the sum of the squares of the differences between the actual values and the predicted values of a model. The primary goal of a regression model is to predict the true values of the dependent variable as accurately as possible. The differences between the predictions and the true values represent the errors. SSE takes the squares of these errors and calculates their sum. Mathematically, for *n* data points, SSE is calculated using the following formula:

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{11}$$

Here:

- *yi* is the true value of the *i*th data point,
- $\hat{y}_i$ is the predicted value of the *i*th data point by the model,
- *n* is the total number of data points.

A low SSE value indicates that the model fits the data better and makes better predictions. As the value of SSE approaches zero, the model's predictions get closer to the true values.

4.2.2. Sensitivity and Specificity

The terms "sensitivity" and "specificity" are used in statistics and medicine to quantitatively characterize how well a test detects the existence or absence of a medical condition. If those with the ailment are viewed as "positive" and people without it as "negative," then a test's sensitivity and specificity may be used to determine whether or not it can accurately detect real positives and true negatives, respectively:

• Sensitivity, often known as the true positive rate, is the likelihood of a positive test result provided the subject is in fact positive.

• Specificity, also known as the true negative rate, is the likelihood of a negative test result, provided that the subject is indeed negative.

Sensitivity and specificity can be specified in reference to an assumed-to-be-correct "gold standard test" if the real state of the ailment is unknown. Sensitivity and specificity are typically traded off in testing, both for diagnosis and screening, such that higher sensitivities imply lower specificities and vice versa. A test will be considered highly sensitive if it can consistently identify the existence of a disease, producing a high proportion of genuine positive results and a low percentage of false negative results.

This is crucial when the ailment has major consequences if left untreated and/or when the treatment is extremely successful with few adverse effects. A high specificity test is one that consistently identifies those who do not have the ailment, producing a high percentage of genuine negative results and a low percentage of false positive results. This is particularly crucial in situations when individuals with a diagnosis may be more likely to undergo tests, incur more costs, experience stigma, worry, etc. Yerushalmy J, introduced the words "sensitivity" and "specificity" to the American biostatistician community [23]. Different definitions exist for laboratory quality control. For instance, according to Saah AJ, Hoover DR, "analytical sensitivity" is the smallest amount of substance in a sample that can be accurately measured by an assay (synonymously to detection limit). "Analytical specificity" is the ability of an assay to measure one specific organism or substance instead of others. However, this essay focuses on the previously mentioned diagnostic sensitivity and specificity [24]. Application for the investigation of screening Consider a research that assesses a test used to check individuals for illnesses. It is either the case that every test taker has the illness or does not. A positive test result would indicate that the subject has the illness, whereas a negative result would indicate that the subject does not. It is possible that the test results do not accurately reflect each subject's current situation. Under those circumstances:

True positive: Ill individuals properly classified as ill

False positive: When healthy individuals are mistakenly classified as ill

False negative: Sick persons were mistakenly recognized as healthy.

True negative: Healthy people were accurately identified as healthy.

The test's sensitivity and specificity may be computed once the figures for true positives, false positives, true negatives, and false negatives have been obtained. Any individual with the illness is likely to be identified by the test as positive if it turns out that the sensitivity is high. Conversely, if the test has a high specificity, it is likely to classify as negative any individual who does not have the condition. The methodology for calculating these ratios is discussed on an NIH website [25].

Sensitivity

Think of a medical test used to diagnose a disease as an example. The capacity of a test to accurately identify sick people among those who actually have the ailment is known as sensitivity, which is also frequently referred to as the detection rate in a clinical environment [26]. Mathematically, this can be expressed as:

$$Sensitivity = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{12}$$

Since a high sensitivity test seldom misdiagnoses patients who actually have the disease, a negative result can be helpful in "ruling out" sickness [26]. All individuals with the condition will be identified by a test that tests positive and has 100% sensitivity. In this instance, a negative test result would categorically rule out the patient's illness. A high sensitivity test result, however, is not always helpful in "ruling in" a condition. Let's say a "phony" test kit is made to consistently provide a positive result. Test sensitivity is 100% when performed on sick individuals, since all of them test positive. False positives are not considered by sensitivity, though. In addition, the fraudulent test has a false positive rate of 100% on all healthy people, meaning that it is ineffective for "ruling in" or identifying the illness. For the purpose of determining sensitivity, indeterminate test results are ignored. Samples that are uncertain can be treated as false negatives, which yield the worst-case sensitivity value and may cause it to be underestimated, or they can be eliminated from the analysis (it is important to specify the number of exclusions when mentioning sensitivity).

Specificity

Examine a medical test as an illustration to help you understand the concept. The capacity of the test to accurately rule out healthy individuals in the absence of a problem is referred to as specificity. Here's how to write it mathematically:

$$Specificity = \frac{TN}{N} = \frac{TN}{TN + FP} \tag{13}$$

Since a test with high specificity seldom yields positive findings in healthy individuals, a positive result might be helpful for "ruling in" illness [27]. A positive test result would categorically rule in the presence of the disease since a test with 100% specificity would identify all patients who do not have the condition by testing negative. However, "ruling out" a condition is not always possible with a negative result from a test with great specificity. Since specificity does not account for erroneous negative results, a test that consistently yields a negative result, for instance, would have a specificity of 100%. Such a test would not be useful for "ruling out" the condition since it would yield a negative result for those who already had the illness.

4.2.3. F1-Score

The F1-score is a performance measure commonly used in classification problems. This metric is used to evaluate the accuracy of a model, particularly useful in imbalanced classification problems. The F1-score is a combined value of the precision and recall metrics. The percentage of positive examples among those that a model predicts to be positive is known as precision. The percentage of genuinely positive cases that the model accurately predicts as positive is known as recall. The F1-score is calculated using the following formula, using precision (P) and recall (R) values:

$$\boldsymbol{F = 2 \cdot \frac{P \cdot R}{P + R}} \tag{14}$$

The F1-score represents a balance between precision and recall. It is particularly used in classification problems where there is an imbalance between classes. The F1-score takes a value between 0 and 1, with a value closer to 1 representing better performance.

5. Results

In this study, the Artificial Algae Algorithm was used to classify 12 datasets (aggregation, banknote, blobs, ecoli, glass, iris, iris2d, ionosphere, seeds, vertebral2, vertebral3 and wine) obtained from the UCI repository. The aim of the algorithms was to identify the clusters that minimize the values of the SSE. The algorithms AAA, ALO, DEA, MFO, PSO, TSA and WOA were run for each dataset with a population size of 40, a maxFEs (maximum fitness evaluation size) of 10000 and a runtime of 30. The values of the F1-Score, Sensitivity and Specificity metrics obtained as a result of this study are shown in the tables below. The three tables below show the F1-Score, Sensitivity and Specificity values obtained in 30 different runs of seven different algorithms for each dataset, as well as the average (Mean) and standard deviation (Std.) of these values. In addition, the "Rank" column is used to indicate the performance of each algorithm. F1-Score, Sensitivity and Specificity are metrics that measure the accuracy of a clustering algorithm, with higher values indicating better performance. ***Note:*** *When writing the results, 2 digits after the comma were taken as precision. Therefore, although the algorithms appear to have the same metric value, their rank may appear different.*

Table 2 shows the results of the algorithms for F1-score metric. According to the Table 2, the AAA algorithm has an average F1-score value of 0.72 for 12 datasets and ranks first in the rank column. The ALO algorithm ranks first in the rank column for the blobs, iris, and iris2d datasets. The MFO and TSA algorithms rank first in the rank column for the banknote, blobs, and iris2d datasets, and the WOA algorithm ranks first in the rank column for the iris2d dataset. In general, Table 2 shows that the AAA algorithm ranks first with an average success ranking of 1, the MFO algorithm ranks second with an average ranking of 2.5, the ALO algorithm ranks third with an average ranking of 2.66, the TSA algorithm ranks fourth with an average ranking of 3.08, the WOA algorithm ranks fifth with an average ranking of 4.16, the DEA algorithm ranks sixth with an average ranking of 4.91, and the PSO algorithm ranks seventh with an average ranking of 5.5.

Table 2. Experimental results of AAA and other algorithms for F1-Score metric

Datasets		**AAA**	**ALO**	**DEA**	**MFO**	**PSO**	**TSA**	**WOA**
Aggregation	Mean	0.81	0.73	0.71	0.77	0.62	0.71	0.67
	Std	0.04	0.10	0.03	0.07	0.14	0.09	0.11
	Rank	1	3	4	2	7	5	6
Banknote	Mean	0.79	0.79	0.46	0.79	0.45	0.79	0.75
	Std	0.00	0.01	0.22	0.00	0.16	0.00	0.08
	Rank	1	2	4	1	5	1	3
Blobs	Mean	1.00	1.00	1.00	1.00	0.83	1.00	0.99
	Std	0.00	0.00	0.00	0.00	0.27	0.00	0.08
	Rank	1	1	2	1	4	1	3
Ecoli	Mean	0.39	0.31	0.29	0.35	0.25	0.29	0.30
	Std	0.11	0.07	0.08	0.12	0.09	0.09	0.09
	Rank	1	3	2	2	7	6	4
Glass	Mean	0.24	0.14	0.14	0.20	0.18	0.18	0.13
	Std	0.06	0.05	0.03	0.06	0.06	0.04	0.03
	Rank	1	5	6	2	3	4	7
Iris	Mean	0.96	0.96	0.83	0.89	0.68	95	0.84
	Std	0.00	0.00	0.09	0.12	0.21	0.01	0.18
	Rank	1	1	5	3	6	2	4
Iris2d	Mean	0.96	0.96	0.94	0.96	0.88	0.96	0.96
	Std	0.00	0.00	0.03	0.00	0.10	0.00	0.00
	Rank	1	1	2	1	3	1	1
Ionosphere	Mean	0.69	0.67	0.40	0.54	0.54	0.67	0.67
	Std	0.00	0.03	0.04	0.13	0.10	0.02	0.05
	Rank	1	4	7	5	6	3	2
Seeds	Mean	0.87	0.83	0.37	0.55	0.29	0.56	0.31
	Std	0.02	0.16	0.15	0.27	0.14	0.25	0.18
	Rank	1	2	5	4	7	3	6
Vertebral2	Mean	0.65	0.53	0.39	0.57	0.41	0.39	0.44
	Std	0.00	0.14	0.02	0.12	0.05	0.13	0.11
	Rank	1	3	6	2	5	7	4
Vertebral3	Mean	0.38	0.21	0.15	0.23	0.15	0.26	0.18
	Std	0.03	0.08	0.04	0.08	0.05	0.08	0.09
	Rank	1	4	6	3	7	2	5
Wine	Mean	0.90	0.39	0.24	0.37	0.31	0.46	0.34
	Std	0.03	0.20	0.15	0.21	0.19	0.14	0.22
	Rank	1	3	7	4	6	2	5
Average Rank		**1**	2.67	4.92	2.5	5.5	3.08	4.17

Table 3 shows the results of the algorithms for Sensitivity metric. According to the Table 3, the AAA algorithm has an average Sensitivity value of 0.75 for 12 datasets and ranks first in the rank column. The ALO algorithm ranks first in the rank column for the blobs, iris, and iris2d datasets. The

MFO and TSA algorithms rank first in the rank column for the banknote, blobs, and iris2d datasets, and the WOA algorithm ranks first in the rank column for the iris2d dataset. In general, Table 3 shows that the AAA algorithm ranks first with an average success ranking of 1, the MFO algorithm ranks second with an average ranking of 2.5, the TSA algorithm ranks third with an average ranking of 2.83, the ALO algorithm ranks fourth with an average ranking of 2.91, the WOA algorithm ranks fifth with an average ranking of 4.25, the DEA algorithm ranks sixth with an average ranking of 5.08, and the PSO algorithm ranks seventh with an average ranking of 5.25.

Table 3. Experimental results of AAA and other algorithms for Sensitivity metric

Datasets		**AAA**	**ALO**	**DEA**	**MFO**	**PSO**	**TSA**	**WOA**
Aggregation	Mean	0.90	0.80	0.77	0.86	0.67	0.78	0.74
	Std	0.06	0.12	0.06	0.09	0.16	0.11	0.12
	Rank	1	3	5	2	7	4	6
Banknote	Mean	0.80	0.80	0.50	0.80	0.53	0.80	0.76
	Std	0.00	0.01	0.20	0.00	0.12	0.00	0.08
	Rank	1	2	5	1	4	1	3
Blobs	Mean	1.00	1.00	1.00	1.00	0.85	1.00	0.99
	Std	0.00	0.00	0.00	0.00	0.24	0.00	0.06
	Rank	1	1	2	1	4	1	3
Ecoli	Mean	0.41	0.37	0.32	0.39	0.28	0.33	0.34
	Std	0.12	0.09	0.09	0.13	0.10	0.10	0.12
	Rank	1	3	6	2	7	5	4
Glass	Mean	0.30	0.22	0.20	0.28	0.24	0.24	0.20
	Std	0.05	0.05	0.02	0.07	0.06	0.04	0.03
	Rank	1	5	7	2	3	4	6
Iris	Mean	0.96	0.96	0.84	0.90	0.72	0.95	0.87
	Std	0.00	0.00	0.07	0.10	0.17	0.01	0.14
	Rank	1	1	5	3	6	2	4
Iris2d	Mean	0.96	0.96	0.94	0.96	0.88	0.96	0.96
	Std	0.00	0.00	0.03	0.00	0.09	0.00	0.00
	Rank	1	1	2	1	3	1	1
Ionosphere	Mean	0.71	0.67	0.50	0.58	0.56	0.68	0.68
	Std	0.00	0.03	0.03	0.11	0.08	0.02	0.04
	Rank	1	4	7	5	6	2	3
Seeds	Mean	0.87	0.84	0.46	0.59	0.40	0.60	0.39
	Std	0.02	0.14	0.12	0.23	0.12	0.22	0.16
	Rank	1	2	5	4	6	3	7
Vertebral2	Mean	0.72	0.59	0.47	0.64	0.48	0.40	0.51
	Std	0.00	0.14	0.05	0.12	0.05	0.16	0.12
	Rank	1	3	6	2	5	7	4
Vertebral3	Mean	0.45	0.34	0.35	0.36	0.35	0.38	0.35
	Std	0.02	0.05	0.03	0.05	0.02	0.05	0.05
	Rank	1	7	4	3	6	2	5
Wine	Mean	0.91	0.54	0.40	0.51	0.44	0.57	0.47
	Std	0.02	0.18	0.13	0.19	0.15	0.13	0.18
	Rank	1	3	7	4	6	2	5
Average Rank		**1**	2.17	5.08	2.5	5.25	2.83	4.25

Table 4 shows the results of the algorithms for Specificity metric. According to the Table 4, the AAA algorithm ranks first in the rank column for 11 datasets and ranks second in the rank column with an average Specificity value of 0.94 for the ecoli dataset. The ALO algorithm ranks first in the rank column for the blobs, ecoli, iris, and iris2d datasets. The MFO and TSA algorithms rank first in the rank column for the banknote, blobs, and iris2d datasets, and the WOA algorithm ranks first in the rank column for the iris2d dataset. In general, Table 4 shows that the AAA algorithm ranks first with an average success ranking of 1.08, the ALO and MFO algorithms rank second with an average ranking of 2.75, the TSA algorithm ranks third with an average ranking of 2.83, the WOA algorithm ranks fourth with an average ranking of 4.41, the DEA algorithm ranks fifth with an average ranking of 4.91, and the PSO algorithm ranks sixth with an average ranking of 5.08.

Table 4. Experimental results of AAA and other algorithms for Specificity metric

Datasets		AAA	ALO	DEA	MFO	PSO	TSA	WOA
Aggregation	Mean	0.98	0.97	0.97	0.97	0.96	0.96	0.96
	Std	0.00	0.01	0.01	0.01	0.02	0.01	0.02
	Rank	1	3	4	2	7	5	6
Banknote	Mean	0.80	0.80	0.50	0.80	0.53	0.80	0.76
	Std	0.00	0.01	0.20	0.00	0.12	0.00	0.08
	Rank	1	2	5	1	4	1	3
Blobs	Mean	1.00	1.00	1.00	1.00	0.92	1.00	0.99
	Std	0.00	0.00	0.00	0.00	0.12	0.00	0.03
	Rank	1	1	2	1	4	1	3
Ecoli	Mean	0.94	0.94	0.92	0.94	0.92	0.93	0.93
	Std	0.01	0.01	0.02	0.02	0.02	0.01	0.02
	Rank	2	1	6	3	7	5	4
Glass	Mean	0.86	0.85	0.84	0.86	0.85	0.85	0.84
	Std	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	Rank	1	5	6	2	4	3	7
Iris	Mean	0.98	0.98	0.92	0.95	0.86	0.98	0.93
	Std	0.00	0.00	0.04	0.05	0.09	0.01	0.07
	Rank	1	1	5	3	6	2	4
Iris2d	Mean	0.98	0.98	0.97	0.98	0.94	0.98	0.98
	Std	0.00	0.00	0.01	0.00	0.05	0.00	0.00
	Rank	1	1	2	1	3	1	1
Ionosphere	Mean	0.71	0.67	0.50	0.58	0.56	0.68	0.68
	Std	0.00	0.03	0.03	0.11	0.08	0.02	0.04
	Rank	1	4	7	5	6	2	3
Seeds	Mean	0.94	0.92	0.73	0.80	0.70	0.80	0.70
	Std	0.01	0.07	0.06	0.12	0.06	0.11	0.08
	Rank	1	2	5	4	6	3	7
Vertebral2	Mean	0.72	0.59	0.47	0.64	0.48	0.40	0.51
	Std	0.00	0.14	0.05	0.12	0.05	0.16	0.12
	Rank	1	3	6	2	5	7	4
Vertebral3	Mean	0.72	0.66	0.67	0.67	0.67	0.68	0.67
	Std	0.01	0.03	0.02	0.03	0.02	0.03	0.03
	Rank	1	7	4	5	3	2	6
Wine	Mean	0.95	0.75	0.70	0.74	0.71	0.76	0.73
	Std	0.01	0.08	0.06	0.08	0.07	0.05	0.08
	Rank	1	3	7	4	6	2	5
Average Rank		**1.08**	2.75	4.92	2.75	5.08	2.83	4.42

When the results are examined in general, it is seen that AAA outperforms other algorithms. This result shows that the local and global search strategies of the algorithm are suitable for the problems studied. In addition, in the position update process of the algorithm, the elimination of bad members and the inclusion of members with better fitness values into the population positively affects its success.

6. Discussion and Conclusion

In this study, the accuracy of clustering was improved by using AAA to obtain appropriate cluster centers and increase clustering success. The performance of AAA was evaluated using 12 commonly used datasets (aggregation, banknote, blobs, ecoli, glass, iris, iris2d, ionosphere, seeds, vertebral2, vertebral3 and wine) from the UCI repository. AAA's performance was compared with the performance of the six metaheuristic algorithms by using three metrics: F1-Score, Sensitivity and Specificity. The experimental results obtained show that AAA is more successful than the other algorithms in the clustering problem.

For future studies, the performance of the algorithms can be increased by using hybrid metaheuristic algorithms. In addition, the performance of the algorithms can be evaluated by using data sets with different characteristics.

References

[1] K. Özdamar, *Paket Programlar İle İstatistiksel Veri Analizi 1*. Kaan Kitabevi, 1997.

[2] H. Tatlıdil, *Uygulamalı Çok Değişkenli İstatistiksel Analiz*. Accessed: Oct. 02, 2024. [Online]. Available: https://www.nadirkitap.com/uygulamali-cok-degiskenli-istatistiksel-analiz-prof-dr-huseyin-tatlidil-kitap1444523.html

[3] J. F. Hair Jr., W. C. Black, B. J. Babin, and R. E. Anderson, *Multivariate Data Analysis*. Accessed: Oct. 02, 2024. [Online]. Available: https://www.scirp.org/reference/ReferencesPapers?ReferenceID=1519308

[4] M. Lorr, *Cluster Analysis for Social Scientists*, 1st edition. San Francisco: Jossey-Bass Inc Pub, 1983.

[5] S. Sharma, *Applied Multivariate Techniques*, 1st edition. New York: Wiley, 1995.

[6] J. Tabak, *Geometry: the language of space and form*, Rev. ed. in The history of mathematics. New York, NY: Facts On File, 2011.

[7] S. Ishak Boushaki, N. Kamel, and O. Bendjeghaba, 'A new quantum chaotic cuckoo search algorithm for data clustering', *Expert Systems with Applications*, vol. 96, Dec. 2017, doi: 10.1016/j.eswa.2017.12.001.

[8] X.-S. Yang, 'A New Metaheuristic Bat-Inspired Algorithm', vol. 284, Apr. 2010, doi: 10.1007/978-3-642-12538-6_6.

[9] A. Karami and M. Guerrero-Zapata, 'A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks', *Neurocomputing*, vol. 149, pp. 1253–1269, Feb. 2015, doi: 10.1016/j.neucom.2014.08.070.

[10] H. Liu and X. Ban, 'Clustering by growing incremental self-organizing neural network', *Expert Systems with Applications*, vol. 42, no. 11, pp. 4965–4981, Jul. 2015, doi: 10.1016/j.eswa.2015.02.006.

[11] M. A. Rahman and M. Z. Islam, 'A hybrid clustering technique combining a novel genetic algorithm with K-Means', *Knowledge-Based Systems*, vol. 71, pp. 345–365, Nov. 2014, doi: 10.1016/j.knosys.2014.08.011.

[12] G. Tzortzis and A. Likas, 'The MinMax *k*-Means clustering algorithm', *Pattern Recognition*, vol. 47, no. 7, pp. 2505–2516, Jul. 2014, doi: 10.1016/j.patcog.2014.01.015.

[13] U. Maulik and S. Bandyopadhyay, 'Genetic algorithm-based clustering technique', *Pattern Recognition*, vol. 33, no. 9, pp. 1455–1465, Sep. 2000, doi: 10.1016/S0031-3203(99)00137-5.

[14] D. Merwe and A. Engelbrecht, 'Data clustering using particle swarm optimization[C]', presented at the Proc of 2003 Congress on Evolutionary Computation (CEC'03), Jan. 2003, pp. 215–220. doi: 10.1109/CEC.2003.1299577.

[15] P. S. Shelokar, V. K. Jayaraman, and B. D. Kulkarni, 'An ant colony approach for clustering', *Analytica Chimica Acta*, vol. 509, no. 2, pp. 187–195, May 2004, doi: 10.1016/j.aca.2003.12.032.

[16] M. Omran, A. Engelbrecht, and A. Salman, 'Dynamic Clustering using Particle Swarm Optimization with Application in Unsupervised Image Classification', vol. 9, Jan. 2005.

[17] C. Zhang, D. Ouyang, and J. Ning, 'An artificial bee colony approach for clustering', *Expert Systems with Applications*, vol. 37, no. 7, pp. 4761–4767, Jul. 2010, doi: 10.1016/j.eswa.2009.11.003.

[18] A. N. Mat, O. İnan, and M. Karakoyun, 'An application of the whale optimization algorithm with Levy flight strategy for clustering of medical datasets', *An International Journal of Optimization and Control: Theories & Applications (IJOCTA)*, vol. 11, no. 2, Art. no. 2, Jun. 2021, doi: 10.11121/ijocta.01.2021.001091.

[19] G. Sariman, 'Veri Madenciliğinde Kümeleme Teknikleri Üzerine Bir Çalışma: K-Means ve K-Medoids Kümeleme Algoritmalarının Karşılaştırılması'.

[20] B. S. Everitt and G. Dunn, *Applied Multivariate Data Analysis*. Oxford University Press, 1992.

[21] S. A. Uymaz, G. Tezel, and E. Yel, 'Artificial algae algorithm (AAA) for nonlinear global optimization', *Applied Soft Computing*, vol. 31, pp. 153–171, Jun. 2015, doi: 10.1016/j.asoc.2015.03.003.

[22] X. Zhang *et al.*, 'Binary Artificial Algae Algorithm for Multidimensional Knapsack Problems', *Applied Soft Computing*, vol. 43, Mar. 2016, doi: 10.1016/j.asoc.2016.02.027.

[23] J. Yerushalmy, 'Statistical problems in assessing methods of medical diagnosis, with special reference to X-ray techniques', *Public Health Rep (1896)*, vol. 62, no. 40, pp. 1432–1449, Oct. 1947.

[24] A. J. Saah and D. R. Hoover, '[Sensitivity and specificity revisited: significance of the terms in analytic and diagnostic language]', *Ann Dermatol Venereol*, vol. 125, no. 4, pp. 291–294, Apr. 1998.

[25] R. Parikh, A. Mathai, S. Parikh, G. Chandra Sekhar, and R. Thomas, 'Understanding and using sensitivity, specificity and predictive values', *Indian J Ophthalmol*, vol. 56, no. 1, pp. 45–50, 2008, doi: 10.4103/0301-4738.37595.

[26] D. G. Altman and J. M. Bland, 'Diagnostic tests. 1: Sensitivity and specificity', *BMJ*, vol. 308, no. 6943, p. 1552, Jun. 1994, doi: 10.1136/bmj.308.6943.1552.

[27] 'SpPin and SnNout'. Accessed: Oct. 02, 2024. [Online]. Available: https://www.cebm.ox.ac.uk/resources/ebm-tools/sppin-and-snnout

Journal of Soft Computing and Artificial Intelligence

Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access

Volume 05
Issue 02

December, 2024

Research Article

Cybersecurity in the Internet of Things: the Detection of the Types of Upcoming Digital Information by Using Classification Techniques

Dima Raed Abu Khalil [1], ***Yousef Abuzir*** [2]

[1,2] *Faculty of Technology and Applied Sciences, Al-Quds Open University, Ramallah, Palestine*

ARTICLE INFO

Article history:
Received October 30, 2024
Revised November 22, 2024
Accepted November 25, 2024

Keywords:
Network Attack Detection
Internet of Things (IoT)
Machine Learning
Logistic Regression
Linear Discriminant Analysis (LDA)
Deep Learning

ABSTRACT

This study addresses the critical challenge of Cyber-attacks detection (CAD) in the Internet of Things (IoT) environment, specifically focusing on the classification of non-malicious and malicious network traffic. The primary objective is to enhance the accuracy and reliability of detection mechanisms through the implementation of advanced machine learning models, particularly the hybrid CNN-GRU-LSTM model. The study utilizes the SYN DoS dataset from the Kitsune Network Attack Dataset to train and evaluate various models, including Linear Discriminant Analysis (LDA), Logistic Regression, and the CNN-GRU-LSTM model. The methodology includes a comprehensive performance analysis of each model, employing metrics such as accuracy, precision, recall, and F1-score. The results reveal that both LDA and Logistic Regression achieved perfect accuracy (1.00), while the CNN-GRU-LSTM model exhibited an accuracy of 0.998. Additionally, the CNN-GRU-LSTM model demonstrated a high area under the curve (AUC) value of 0.8559, indicating strong discriminatory power. The study further employs SHAP (SHapley Additive exPlanations) for model interpretability, allowing for a detailed analysis of feature importance and insights into model behavior. In conclusion, the hybrid CNN-GRU-LSTM model offers a promising approach for effective network attack detection while providing a basis for future improvements in real-time applications and the exploration of additional datasets.

1. Introduction

In the era of digital transformation, network security has become a critical concern for organizations worldwide. With the increasing reliance on networked systems, the frequency and sophistication of cyberattacks have escalated, posing significant threats to data integrity, privacy, and operational stability [1,2]. Network attacks, including Distributed Denial of Service (DDoS) and Man-in-the-Middle (MitM) attacks, can disrupt services and compromise sensitive information, making it imperative to develop robust methods for detecting and classifying these threats effectively [3, 4].

Traditional network attack detection methods, often reliant on signature-based detection, struggle to keep pace with the evolving nature of cyber threats. Attackers continuously develop new methods to

* Corresponding author
e-mail: yabuzir@qou.edu
DOI: 10.55195/jscai.1576195

bypass signature-based defenses, highlighting the limitations of these traditional approaches. False positives and negatives further hamper the effectiveness of these methods, leading to wasted resources and potential security breaches [5] [6] [7].

Deep learning (DL), a subfield of artificial intelligence, offers a promising solution for network attack detection [9]. DL algorithms are effective at identifying complex patterns within vast amounts of dataset. By examining network traffic data, DL models have the ability learn to distinguish between normal and malicious network behavior, offering a more adaptable and robust approach to network attack detection [10] [11] [12].

This research explores the potential of deep learning (DL) and machine learning (ML) techniques for accurately predicting and detecting SYN DoS attacks. We hypothesize that deep learning models can outperform traditional methods due to their superior pattern recognition capabilities [11]. We will investigate the effectiveness of two machine learning and a deep learning hybrid CNN-GRU-LSTM model – for identifying SYN DoS attacks within the Kitsune SYN DoS dataset (https://www.kaggle.com/datasets/ymirsky/network-attack-dataset-kitsune/data). Our evaluation will compare the performance of these machine and deep learning model, emphasizing key metrics such as accuracy, precision, recall, and F1-score [12] [13].

The Problem Question is How can we accurately classify and detect various types of network attacks using machine and deep learning techniques, and what are the most effective models for distinguishing between harmful and benign network traffic?

The motivation behind this research stems from the growing need for advanced and efficient solutions to enhance network security. Traditional methods of network attack detection often struggle with high false-positive rates and limited adaptability to new attack vectors. Machine learning offers the potential to improve detection accuracy and adapt to evolving threats by learning from historical data. By exploring and comparing various machine learning models, this research aims to identify the most effective approaches for classifying network attacks, thereby contributing to the development of more resilient cybersecurity systems.

The novelty of this study lies in the integration of CNN, GRU, and LSTM into a single hybrid model that addresses the complexities of network traffic data. Previous studies have largely relied on individual models, such as CNNs for feature extraction or LSTMs for sequence learning, but have not explored the synergy between these architectures in the context of network attack detection.

This study makes several key contributions to the field of network security and machine learning:

- Comprehensive Evaluation of Models: The study assesses and compares the performance of Linear Discriminant Analysis (LDA), Logistic Regression, and a hybrid CNN-GRU-LSTM modelwithin the framework of network attack classification. A comprehensive evaluation of this hybrid approach, demonstrating its superior accuracy and ability to handle class imbalance in detecting both malicious and benign network instances.
- Detailed Analysis of Network Attack Data: By using the Kitsune Network Attack Dataset, the study provides an in-depth analysis of various attack types, including SYN DoS, and demonstrates how machine learning models can be applied to detect and classify these attacks.
- Model Interpretability: The use of SHAP (SHapley Additive exPlanations) to explain the models' predictions provides valuable information about the factors influencing the classification decisions, enhancing the transparency and trustworthiness of the machine learning models.
- The hybrid model that leverages the strengths of CNN for spatial feature extraction, GRU for short-term temporal dependencies, and LSTM for capturing long-term temporal dependencies in network traffic.
- The integration of advanced optimization techniques such as dropout regularization and early stopping to avoid overfitting and ensure model generalization on unseen data.

This hybrid model provides a novel framework for intrusion detection systems, improving both accuracy and computational efficiency compared to traditional and single-model approaches.

This study examines the development of machine learning for network attack detection. It begins by reviewing existing methods and datasets before delving into model development using techniques like LDA, logistic regression, and hybrid CNN-GRU-LSTM. Model performance is evaluated using standard metrics, and their effectiveness is analyzed using interpretability tools like SHAP. The study concludes by discussing the implications of the findings for network security and outlining directions for future research.

2. Literature Review

Researchers are increasingly interested in using deep learning (DL) to create online network attack detection. This is because machine learning (ML) and DL techniques have been shown to be effective in identifying cyberattacks launched from compromised Internet of Things (IoT) devices [14], [15].

One challenge with traditional ML-based network attack detections is that they require a lot of labelled data for training. To address this, a new approach called Decentralized and Online Federated Learning Intrusion Detection (DOF-ID) has been proposed. This system allows collaborating devices to share information and improve intrusion detection performance across the network [16] [17].

Another promising technique is Deep Transfer Learning (DTL), a type of DL that can be used to enhance intrusion detection within industrial control systems. DTL allows the system to gain insights from data in one domain (such as general network traffic) and apply insights to a different domain (such as industrial control systems) [18] [19]. This can improve detection accuracy and help mitigate threats more effectively. Overall, these studies show that advanced ML and DL techniques have a lot of potential for improving the capabilities of online network intrusion detection systems.

In the paper by Hussain et al. (2023), the authors reviewed various intrusion detection models and the threats posed to IoT systems by compromised devices, emphasizing the use of ML and DL techniques as effective defensive measures [15]. Mert et al. (2023) proposed the DOF-ID architecture, which enhances intrusion detection by allowing IDSs to learn from both local and remote data while maintaining data privacy, showing significant performance improvements on Kitsune and Bot-IoT datasets [20]. Kheddar et al. (2023) provided a comprehensive review of using deep transfer learning (DTL) in industrial control networks for intrusion detection, highlighting improvements in detection accuracy and the use of multiple datasets and evaluation metrics like accuracy and false alarm rate [18].

Wasnik and Chavhan (2023) tested different DL algorithms on public malware data. They found these models work well, but need frequent updates to stay ahead of changing attack patterns. The authors propose a specific deep neural network (DNN) that can adapt to dynamic network behavior. They suggest constantly improving these models and linking them to real-time monitoring for proactive threat detection [20]. Ogundokun et al. (2023) systematically reviewed the application of ML and DL algorithms in IDS, analyzing various classifiers, datasets, and frameworks used from 2016 to 2021, and offering insights into recent advancements and challenges [22].

Krishna et al. (2020) focused on building an IDS that can also prevent attacks (DOS, Probe, R2L, U2R). They used a Multi-Layer Perceptron (MLP) deep learning model on the KDDCup99 dataset and achieved high accuracy against different attack types. Their system combines detection and prevention, proving effective in real-time situations. The authors recommend further development of the prevention mechanisms and testing the system across diverse network environments [23].

The study of (Fadel et al., 2022) proposed the HDLIDP framework, combining signature-based and deep learning techniques to improve DDoS attack detection and prevention in SDNs, demonstrating significant accuracy improvements through experiments on traditional and SDN datasets [24].

The research by (Alghamdi, 2022) proposes a hybrid intrusion detection model (PO-CFNN) for securing Internet of Things (IoT) devices. It uses a unique optimization technique and achieves high performance on test datasets. The author suggests adapting the model for more complex IoT scenarios and improving its efficiency [25].

The review by (Monani et al., 2023) explores different ways to analyze cyber threats using data, like predicting denial-of-service attacks. The study suggests combining these methods for stronger defenses [26].

Deep Learning for Network Security: The survey of (Auwal, 2022) highlights how deep learning can improve network intrusion detection compared to traditional methods. It identifies areas for further research, like exploring techniques that don't require as much labeled data [27].

The study of (Hnamte & Hussain, 2023) proposes a deep learning model using convolutional neural networks to detect and classify network intrusions. It shows promising results, but needs testing with more diverse data to ensure real-world effectiveness [28].

Alabdulatif and Rizvi (2022) addressed the improvement of Kitsune Network Intrusion Detection (NID) using machine learning techniques. They evaluated various tree algorithm variants on Kitsune datasets, ultimately recommending the Fine Tree algorithm for better performance. The main metrics used were effectiveness and efficiency. The results indicated that the Fine Tree algorithm outperformed other tree variants in terms of improving Kitsune NID's accuracy and reliability [29].

The study by Malliga et al. (2022) examines the efficacy of deep learning techniques in identifying DoS/DDoS attacks. They concluded that deep learning is capabile of managing these evolving threats [30]. Sujatha et al. (2023) examined the application of deep reinforcement learning (DRL) for network intrusion detection [31]. They reported that their DQL model achieve high degree and impressive accuracy in identifying intrusion.

Mohammed et al. (2023) in their review on machine learning and deep learning stratigies for DDoS detection in Software-Defined Networking (SDN) frameworks. Their results reveal highlight a growing interest in utilizing these techniques, with challenges related to datasets [32]. In a similar vein, Omarov et al. (2022) explore current techniques for detecting network intrusions in Internet of Things (IoT) scenarios. They point out the lack of computational models and formal justification of attacks as key challenges in this area [33].

Researchers are exploring various Machine Learning (ML) and Deep Learning (DL) techniques to improve cyberattack detection. These techniques address challenges like data privacy, adaptability, and evolving threats. Studies recommend ongoing model refinement, using advanced algorithms, and testing in real-world scenarios. Overall, these advancements in ML and DL are crucial for building robust and adaptable cyberattack detection systems to combat cyber threats.

Table 1 Summary for some researches in applying ML and DL in network attack detection

Reference	**Problem**	**ML/DL Techniques**	**Evaluation Metrics**	**Main Results**
Hussain et al. (2023)	IoT-based cyber-attacks due to device proliferation	ML, DL	Not specified	Effective control against IoT-originated attacks
Mert et al. (2023)	Limited applicability of ML-based IDSs due to private local data	Federated learning	Accuracy, computation time	Improved intrusion detection performance across nodes
Kheddar et al. (2023)	Protecting industrial control systems from various threats	Deep Transfer Learning	Accuracy, F-score, false alarm rate	Enhanced IDS performance with scarce labeled data
Ogundokun et al. (2023)	Lack of comprehensive studies on ML for IDS	ML, DL algorithms	Not specified	Insights into advancements and challenges from 2016-2021
Fadel et al. (2022)	DDoS attacks on SDN controllers	Hybrid Deep Learning	Classification accuracy	Significant improvement in detection accuracy
Alabdulatif and Rizvi (2022)	Improving Kitsune NID	Variants of Tree algorithms (Simple Tree, Medium Tree, Coarse Tree, RUS Boosted, Bagged Tree)	Confusion Matrix, Speed, Accuracy	Fine Tree algorithm outperformed others
Malliga, Nandhini, Kogilavani (2022)	Detecting DoS/DDoS attacks	Deep learning models	Various performance metrics	Deep learning models have improved detection capabilities but need further enhancement

Sujatha et al. (2023)	Network intrusion detection	Deep Q-Learning (DQL)	Accuracy, recall rate, precision	DQL model achieved 91.4% accuracy, outperforming other models
Bahashwan et al. (2023)	Detecting DDoS attacks in SDN	ML, DL, hybrid approaches	Evaluation based on Various performance metrics (Accuracy) and unrealistic datasets	Ensemble, hybrid, and single ML-DL approaches are most used but need improvement
Omarov et al. (2022)	Network intrusion detection	Taxonomy of detection technologies	Evaluation of advanced research topics	Highlights need for computational models and formal attack justification

3. Methodology

In this study, we implemented a systematic and robust methodology to address the challenges posed by network attack detection using machine learning techniques. Our approach involves several key stages, from data acquisition and preprocessing to model training, evaluation, and interpretability. The goal is to develop effective and interpretable models that can accurately classify network traffic as benign or malicious. Figure 1, presents a conceptual diagram illustrating the methodology for network attack detection research. The diagram highlights the workflow from data acquisition through model evaluation and interpretability.

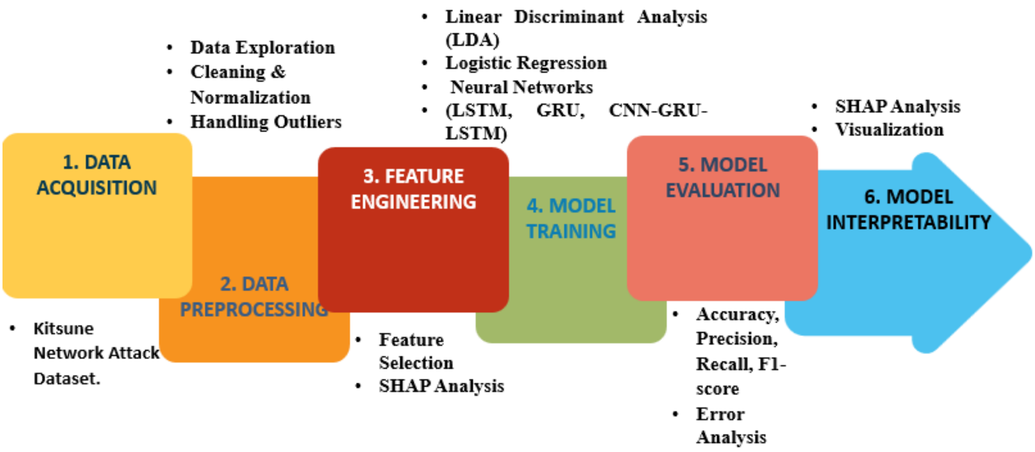


Figure 1 A conceptual diagram illustrating the methodology

Data Acquisition: The first step involves sourcing the Kitsune Network Attack Dataset, specifically the SYN DoS dataset. It provides a rich set of features related to various types of network attacks. This dataset serves as the foundation for our study, offering diverse and representative samples for model training and evaluation.

Data Preprocessing: Once the data is acquired, it undergoes extensive preprocessing. This stage includes data exploration to understand the structure and quality of the dataset, followed by cleaning and normalization to prepare the data for model training. Outlier detection and handling are also performed to ensure the data's integrity and improve model performance.

Feature Engineering: Feature engineering is a crucial step where we select and refine the features used in model training. This step applies techniques such as SHAP (SHapley Additive exPlanations) to assess feature significance and comprehend the impact of different features on model predictions. This step ensures that the models leverage the most relevant information for accurate classification.

Model Training: We then train a variety of machine learning models to address the classification task. The models include Linear Discriminant Analysis (LDA), Logistic Regression, and a hybrid CNN-GRU-LSTM model. Each model is trained and fine-tuned to optimize its performance in detecting network attacks.

Model Evaluation: The trained models are evaluated using several metrics to assess their performance. Metrics such as accuracy, precision, recall, and F1-score are calculated to provide a comprehensive view of each model's effectiveness. Error analysis is also conducted to identify any patterns in misclassification and areas for improvement.

Model Interpretability: To ensure that the models are not only effective but also interpretable, we use SHAP analysis to explain model predictions. This step involves visualizing SHAP values to understand how different features influence the model's decisions, providing transparency and trust in the model's outputs.

The methodology outlined here integrates these steps into a coherent process aimed at developing robust and interpretable models for network attack detection. By systematically addressing each phase of the study, we aim to enhance the reliability and practical applicability of machine learning solutions in network security.

3.1 Dataset and Preprocessing

3.1.1. Description of the Kitsune Network Attack Dataset

The Kitsune Network Attack Dataset is a comprehensive dataset designed to facilitate the analysis and classification of various network attacks. It includes network traffic data captured from a simulated environment where different types of network attacks were introduced. The dataset is hosted on Kaggle and provides CSV files with detailed information about network traffic, including both benign and malicious activities.

The dataset encompasses a wide range of attack types, such as:

- ARP MitM (Address Resolution Protocol Man-in-the-Middle)
- SYN DoS (SYN Denial of Service)
- Active Wiretap

Each type of attack is represented by specific CSV files containing attributes related to the network traffic during the attack. Key attributes typically include time-related features, packet counts, and other metrics essential for understanding the nature and impact of the attacks.

The research utilized the SYN DoS dataset from the Kitsune Network Attack Dataset to analyze and evaluate network attack detection systems. This dataset specifically focuses on SYN flood attacks, a common type of denial-of-service attack that targets the TCP handshake process. By leveraging the SYN DoS dataset, the study aims to develop and test detection algorithms that can effectively identify and mitigate such attacks, thereby enhancing network security measures.

3.1.2. Data Exploration and Preprocessing Steps

Data Inspection: Initial exploration of the dataset involves using methods such as df.info(), df.head() (Figure 2), and df.describe() to understand the structure, size, and summary statistics of the dataset. The output in Figure 2 the intial five rows and first 115 columns of the SYN DoS dataset, are presented, as well as relevant information about the DataFrame. Figure 3, shows a summary statistics table for the DataFrame df. By default, it calculates the count, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values for each numeric column in the DataFrame. The output shows summary statistics for all 115 columns in the DataFrame. The first column, Count, shows the number of non-null values in each column. The Mean column shows the average value for each column, and the Standard column shows the standard deviation. The lowest column displays the minimum value, and the 25% column displays the 25th percentile. The 50% column displays the average, and the 75% column displays the 75th percentile. Finally, the Maximum column displays the maximum value for each column. Here we notice that the count values for all columns are the same, which indicates that there are no missing values in the DataFrame. In addition, the mean values for columns 107-115 are very small, indicating that these columns may contain mostly zero values.

These methods provide insights into data types, missing values, and the basic distribution of feature values.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2771276 entries, 0 to 2771275
Columns: 115 entries, 1 to 115
dtypes: float64(115)
memory usage: 2.4 GB
```

	1	2	3	4	5	6	7	8	9	10	...	106	107	108	109	110	111	112	113	114	115
0	1.000000	1450.000000	0.000000	1.000000	1450.000000	0.000000	1.000000	1450.000000	0.000000	1.000000	...	0.000000e+00	0.0	0.0	1.000000	1450.0	0.000000	1450.0	0.000000e+00	0.0	0.0
1	1.999868	773.955314	456975.998003	1.999921	773.973188	456975.999281	1.999974	773.991063	456975.999920	1.999997	...	0.000000e+00	0.0	0.0	1.000000	98.0	0.000000	98.0	0.000000e+00	0.0	0.0
2	2.998371	999.425957	406159.137822	2.999023	999.388902	406175.842945	2.999674	999.351854	406192.541806	2.999967	...	9.313226e-10	0.0	0.0	1.999998	1450.0	0.000022	1450.0	4.656613e-10	0.0	0.0
3	3.998029	773.958387	456975.998268	3.998817	773.975032	456975.999377	3.999606	773.991677	456975.999931	3.999961	...	0.000000e+00	0.0	0.0	1.999998	98.0	0.000000	98.0	0.000000e+00	0.0	0.0
4	4.995328	909.293168	438671.758562	4.997196	909.255891	438681.843992	4.999065	909.218627	438691.922949	4.999906	...	4.656613e-10	0.0	0.0	2.999995	1450.0	0.000022	1450.0	4.656613e-10	0.0	0.0

5 rows × 115 columns

Figure 2 Output of df.info() and df.head().

	1	2	3	4	5	6	7	8	9	10
count	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06
mean	6.211946e+01	8.024928e+02	3.053892e+05	9.963715e+01	8.032211e+02	3.055215e+05	2.913213e+02	8.041999e+02	3.056160e+05	2.881085e+03
std	2.558334e+01	3.599505e+02	1.805006e+05	3.631008e+01	3.590755e+02	1.798311e+05	9.693310e+01	3.582465e+02	1.792374e+05	9.339921e+02
min	1.000000e+00	6.000000e+01	0.000000e+00	1.000000e+00	6.000000e+01	0.000000e+00	1.000000e+00	6.000000e+01	0.000000e+00	1.000000e+00
25%	4.541724e+01	7.051726e+02	5.651378e+04	7.884313e+01	7.059765e+02	5.569725e+04	2.417702e+02	7.075055e+02	5.464492e+04	2.252462e+03
50%	5.884923e+01	7.336919e+02	4.165450e+05	9.593812e+01	7.348747e+02	4.156597e+05	2.769001e+02	7.367504e+02	4.145451e+05	2.719014e+03
75%	7.517996e+01	7.509993e+02	4.308875e+05	1.172007e+02	7.488586e+02	4.301650e+05	3.672178e+02	7.466790e+02	4.296456e+05	3.787441e+03
max	2.683355e+02	1.452000e+03	4.662590e+05	3.107419e+02	1.452000e+03	4.686412e+05	7.215048e+02	1.452000e+03	4.725568e+05	5.610049e+03

8 rows × 115 columns

...	106	107	108	109	110	111	112	113	114	115
...	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06	2.771276e+06
...	3.876350e+04	1.404329e-05	5.653657e-08	1.704882e+04	8.048290e+02	1.455306e+02	9.230243e+02	3.916362e+04	1.278992e-06	1.765209e-08
...	2.932333e+04	7.530482e-03	2.523086e-05	6.486149e+03	6.313614e+02	1.187150e+02	6.062710e+02	2.912478e+04	6.862339e-04	8.380853e-06
...	0.000000e+00	-5.891966e-11	-3.689219e-07	1.000000e+00	6.000000e+01	0.000000e+00	6.000000e+01	0.000000e+00	-1.036595e-10	-4.496757e-07
...	8.847564e-09	0.000000e+00	0.000000e+00	1.155145e+04	1.860487e+02	1.100327e-04	1.907607e+02	2.281740e-08	0.000000e+00	0.000000e+00
...	4.371386e+04	0.000000e+00	0.000000e+00	1.618504e+04	1.395211e+03	2.079929e+02	1.399811e+03	4.525933e+04	0.000000e+00	0.000000e+00
...	5.747255e+04	0.000000e+00	0.000000e+00	2.177835e+04	1.450000e+03	2.381823e+02	1.450000e+03	5.673080e+04	0.000000e+00	0.000000e+00
...	1.672101e+05	5.532524e+00	1.502178e-02	2.705802e+04	1.452000e+03	4.089046e+02	1.452000e+03	1.672029e+05	4.614913e-01	4.624338e-03

Figure 3 Summary of the distribution of data in a DataFrame using df.describe().

To effectively analyze and evaluate network attack detection systems using the SYN DoS dataset from the Kitsune Network Attack Dataset, a series of preprocessing steps are undertaken.

- Missing Values and Data Integrity: The dataset is checked for missing values using the isnull() function. Figure 4, shows that there are no missing values in the DataFrame df. Each column has 0 missing values, shown as 0 for each column. Columns with missing data are identified and handled appropriately, either by imputing missing values or removing columns or rows with excessive missing data.
- Feature Scaling: Features are scaled to ensure that all variables contribute equally to the model training process. Standard scaling (mean = 0, variance = 1) or Min-Max scaling (rescaling to a range of 0 to 1) is applied depending on the nature of the features and the requirements of the machine learning models used.
- Data Splitting: Using the train_test_split function from sklearn.model_selection,The dataset is split into training and testing. This confirms that the model is evaluated on unseen data, providing an unbiased assessment of its performance.

```
1      0
2      0
3      0
4      0
5      0
      ..
111    0
112    0
113    0
114    0
115    0
Length: 115, dtype: int64
```

Figure 4 Demonstrates that the DataFrame df contains no missing values.

3.1.3. Handling Outliers and Feature Engineering

1 **Outlier Detection and Removal**: Outliers are identified using the Interquartile Range (IQR) method. This involves calculating the IQR for each feature and removing data points that fall outside the range defined by 1.5 times the IQR above the third quartile (Q3+1.5×IQR) and below the first quartile (Q1−1.5×IQR). Removing outliers helps in improving the model's robustness and preventing skewed results. After filtering, the logistic regression model is applied to this refined dataset, which enhances the reliability of the analysis and

ensures that the predictions made by the model are based on more accurate data.

2 **Feature Engineering**: Feature engineering involves creating new features or modifying existing ones to enhance the model's predictive power. In this dataset:

- Aggregation: New features may be created by aggregating raw packet counts into statistical measures such as mean, variance, or frequency of specific attack patterns.
- Normalization: Certain features may be normalized to bring all values within a common scale, making it easier for machine learning algorithms to converge.
- Dimensionality Reduction: Approaches like Principal Component Analysis (PCA) might be used to lessen the number of features while preserving most of the data's variance. This helps in managing computational complexity and potentially improving model performance.

3 Feature Selection: Feature selection involves identifying the most pertinent features for the classification task. Utilizing methods like correlation analysis (using df.corr()) aid in exploring the relationships between features and selecting those that have the most substantial effect on the target variable. Redundant or highly correlated features may be removed to simplify the model and reduce overfitting. The code correlation = df.corr() calculates the pairwise correlation coefficients between all columns in the DataFrame df and stores the result in a new DataFrame correlation. This is the correlation matrix of the df DataFrame. In Figure 5, Each cell in the matrix represents a Pearson correlation coefficient between two columns in the DataFrame. All diagonal elements are 1, because the correlation of the column with itself is always 1.

A correlation matrix can help to identify which columns in a DataFrame are closely associated with one another. For instance, columns 2 and 112 showa correlation coefficient of 0.19972, reflecting a weak positive correlation. In contrast, columns 2 and 115 have a correlation coefficient of -0.002829, indicating a very weak negative correlation.

It is vital to understand that correlation relationship, does not necessarily mean that one variable influences the other (not causation). Additionally, important to note that the correlation matrix only captures linear relationships between variables, so it may not detect nonlinear relationships.

	1	2	3	4	5	6	7	8	9	10	...	106	107	108	109	110	111	112	113	114	115
1	1.000000	0.211223	0.433480	0.972390	0.206749	0.429156	0.863599	0.201641	0.423532	0.766574	...	-0.090311	-0.004296	-0.004660	0.300904	0.120747	0.104893	-0.158262	-0.084449	-0.004292	-0.004642
2	0.211223	1.000000	-0.292155	0.237334	0.999843	-0.291028	0.256526	0.999178	-0.290459	0.261383	...	0.043697	-0.001912	-0.002875	0.701733	0.576333	0.404360	0.199720	0.036699	-0.001877	-0.002879
3	0.433480	-0.292155	1.000000	0.485411	-0.290553	0.999680	0.525095	-0.288794	0.998359	0.530872	...	-0.122740	-0.003047	-0.003184	-0.347439	-0.158190	0.014799	-0.494454	-0.112105	-0.003042	-0.003011
4	0.972390	0.237334	0.485411	1.000000	0.234132	0.482801	0.953599	0.230051	0.478867	0.875378	...	-0.095476	-0.004883	-0.005235	0.349405	0.132438	0.125231	-0.184080	-0.089563	-0.004878	-0.005216
5	0.206749	0.999843	-0.290553	0.234132	1.000000	-0.289330	0.255614	0.999702	-0.288621	0.261787	...	0.048018	-0.002007	-0.003141	0.702458	0.572486	0.409412	0.194342	0.041129	-0.001968	-0.003129
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
111	0.104893	0.404360	0.014799	0.125231	0.409412	0.017914	0.142006	0.414131	0.020471	0.146906	...	0.854485	-0.001277	-0.001307	0.480454	-0.482740	1.000000	-0.740928	0.876133	-0.001256	-0.001330
112	-0.158262	0.199720	-0.494454	-0.184080	0.194342	-0.497337	-0.204097	0.188876	-0.499964	-0.208635	...	-0.690959	0.001476	0.001774	-0.003781	0.812327	-0.740928	1.000000	-0.717059	0.001475	0.001668
113	-0.084449	0.036699	-0.112105	-0.089563	0.041129	-0.109823	-0.094225	0.045129	-0.107952	-0.097041	...	0.976118	0.000387	0.000467	0.269599	-0.721399	0.876133	-0.717059	1.000000	0.000387	0.000410
114	-0.004292	-0.001877	-0.003042	-0.004878	-0.001968	-0.002949	-0.005282	-0.003393	-0.002259	-0.003816	...	0.000636	0.993500	0.791898	-0.000594	-0.000985	-0.001256	0.001475	0.000387	1.000000	0.836281
115	-0.004642	-0.002879	-0.003011	-0.005216	-0.003129	-0.002927	-0.005459	-0.003935	-0.002170	-0.003850	...	0.000754	0.833514	0.994417	-0.000334	-0.000990	-0.001330	0.001668	0.000410	0.836281	1.000000

115 rows × 115 columns

Figure 5 Represents a Pearson correlation coefficient between columns in the DataFrame

Encoding Categorical Variables: If the dataset includes categorical variables, these are encoded into numerical values using techniques like one-hot encoding or label encoding. This ensures that the machine learning models can process all types of data effectively.

Through these preprocessing steps, the dataset can be prepared for robust machine learning analysis, ensuring that the models trained on this data are both accurate and reliable.

4. Machine Learning Models

This section covers the materials and tools used in the study, including libraries, model architectures, and computational resources. It should provide the technical depth, particularly in terms of describing the methods, tools, and resources in more detail.

4.1. Tools and Libraries

For building and evaluating the machine learning models, the following tools and libraries were employed:

- **TensorFlow/Keras**: Used for designing and training the neural network models, including the CNN, GRU, and LSTM components. These libraries provide high-level APIs for deep learning model construction, optimization, and evaluation.
- **Scikit-learn**: Used for implementing classical machine learning models like Logistic Regression and Linear Discriminant Analysis (LDA). Additionally, Scikit-learn was used for model evaluation and metrics calculation, such as accuracy, precision, recall, and F1-score.
- **SHAP** (SHapley Additive exPlanations): Utilized for model interpretability. SHAP values were used to explain the impact of individual features on the model's predictions, helping to understand the factors driving the classification results.
- **Matplotlib/Seaborn**: These Python libraries were used for visualizing model performance, including confusion matrices, ROC curves, and SHAP value plots.
- **Pandas**: Used for data preprocessing, including feature extraction, cleaning, and data splitting into training and test sets.
- **NumPy**: Essential for handling large numerical data arrays, especially in the context of deep learning and time-series analysis.
- **GPU** Resources: Models were trained using GPU-enabled instances (e.g., NVIDIA Tesla P100) to accelerate the training process of deep neural networks, which typically require substantial computational power.

4.2. Model Architectures and Hyperparameters

The machine learning models implemented in this study include:

1. **Linear Discriminant Analysis (LDA)**: A statistical method used for classification based on finding linear decision boundaries between classes.
   - **Key Parameters**: Regularization method (shrinkage), solver, and priors.
2. **Logistic Regression**: A simple yet effective algorithm for binary classification tasks, utilized as a baseline in our model comparison.
   - **Key Parameters**: Regularization type (L2), solver (saga), and learning rate.
3. **Hybrid CNN-GRU-LSTM Model [34]**: The hybrid model integrates three components—**Convolutional Neural Networks (CNN)**, **Gated Recurrent Units (GRU)**, and **Long Short-Term Memory networks (LSTM)**—to handle both spatial and temporal features of network traffic data.
   - **CNN Architecture**:
     - Layers: Convolutional layers followed by pooling layers.
     - Activation Function: ReLU.
     - Regularization: Dropout (0.3), batch normalization.
     - Optimizer: Adam optimizer with learning rate decay.
   - **GRU Architecture**:
     - Layers: Stacked GRU units to model sequential dependencies.
     - Activation Function: Tanh.
     - Regularization: Dropout (0.3).
   - **LSTM Architecture**:
     - Layers: LSTM units with time-sequence dependency modeling.
     - Activation Function: Tanh.
     - Regularization: Dropout (0.3).
   - **Hyperparameters**: The hybrid model was trained for **50 epochs**,

using a **batch size of 32** and **Adam optimizer** with an initial learning rate of 0.001. Early stopping with a patience of 5 epochs was employed to prevent overfitting.
  - **Dropout**: Applied to both GRU and LSTM layers to mitigate overfitting.

4.3. Computational Resources

The training of deep learning models (CNN, GRU, LSTM, and hybrid models) required substantial computational resources:

- **Hardware**: **NVIDIA Tesla P100 GPUs** were used for faster model training, given the complexity and high computational requirements of deep learning models.
- **Cloud Infrastructure**: Models were trained on cloud computing platforms (e.g., **Google Cloud Platform** and **Amazon Web Services (AWS)**) to access GPU resources on-demand and facilitate parallel processing during training.

4.4. Evaluation Metrics

The performance of the models was assessed using a variety of evaluation metrics, including:

- **Accuracy**: Measures the overall percentage of correct predictions.
- **Precision and Recall**: Precision (True Positives / (True Positives + False Positives)) and Recall (True Positives / (True Positives + False Negatives)) were evaluated for both the benign and malicious classes to understand the model's ability to correctly classify each type of traffic.
- **F1-Score**: The harmonic mean of precision and recall, used to balance both metrics in the case of imbalanced datasets.
- **ROC Curve and AUC**: The Receiver Operating Characteristic curve was plotted, and the **Area Under the Curve (AUC)** was calculated to evaluate the model's ability to discriminate between benign and malicious traffic.
- **Mean Squared Error (MSE)**: Used for regression-based evaluation, where applicable.
- **SHAP Values** [40]: Applied to interpret the contribution of each feature in the final model prediction, providing insights into which features (e.g., packet size, source IP address) are most influential in identifying malicious behavior.

4.5. Overview of Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a supervised classification technique aimed at finding a linear combination of features that optimally distinguishes and separates multiple or different classes of objects or events. The primary goal of LDA is dimensionality reduction while preserving as much class discriminatory information as possible. LDA works by projecting the data onto a lower-dimensional space, which maximizes the distance between the means of different classes and minimizes the variance within each class. This projection is particularly useful in scenarios where the number of features is much greater than the number of samples. In network attack classification, LDA can aid and facilitate the identification of malicious traffic from legitimate traffic, enhancing effective classification [34].

The dataset is split into training and testing sets with the help of the train_test_split function from sklearn.model_selection. Subsequently, the LDA model is fitted to the training set and employed to predict labels for both the training and test sets. Figure 6 illustrates a classification results and accuracy analysis report for both the training and test datasets. The report includes precision, recall, and F1 score for each category.

```
--- training
              precision    recall  f1-score   support

not malicous       1.00      1.00      1.00   2073170
    malicous       0.92      0.25      0.39      5287

    accuracy                           1.00   2078457
   macro avg       0.96      0.62      0.69   2078457
weighted avg       1.00      1.00      1.00   2078457

0.9980288261917374
--- test data
              precision    recall  f1-score   support

not malicous       1.00      1.00      1.00    691068
    malicous       0.90      0.24      0.38      1751

    accuracy                           1.00    692819
   macro avg       0.95      0.62      0.69    692819
weighted avg       1.00      1.00      1.00    692819

0.9980225715518772
```

Figure 6 LDA Model Performance Summary

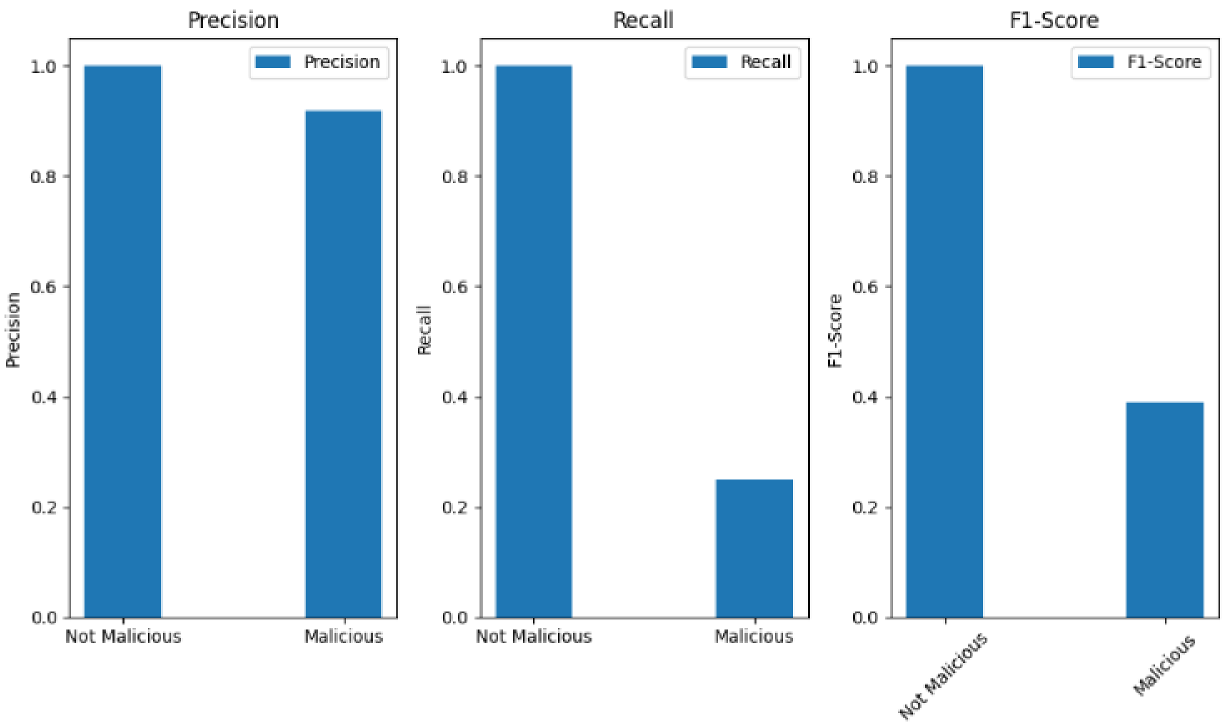


Figure 7 Plot LDA Model Performance Summary

The Mean Squared Error (MSE) for the training data is 0.0019711738082625716 and for the testing data the error is 0.0019774284481228143, as illustrated in (Figure 8). This indicates that the LDA model's predictions for both the training and testing data are generally very close to the actual values, demonstrating a low error rate.

Training MSE = 0.0019711738082625716
Testing MSE = 0.0019774284481228143

Figure 8 The Mean Squared Error (MSE) for the training data and for the testing data

4.6. Logistic Regression and Its Application

Logistic Regression is a statistical model used for binary classification problems. It estimates the probability of a binary outcome based on one or more predictor variables. The logistic function (or sigmoid function) is used to model the relationship between the dependent variable and independent variables, providing outputs in the range (0, 1) that can be interpreted as probabilities. In network security, Logistic Regression is employed to classify network traffic as either benign or malicious based on various features extracted from the traffic data [41]. This method is valued for its simplicity and interpretability, making it suitable for initial explorations and benchmarking against more complex models. In this case, multiclass (multi_class='ovr') and solver 'lbfgs' are used to handle the solution, and 11 job jobs (n_jobs=11) are used to speed up training. The results of the Logistic Regression model provide a detailed view of its performance across various metrics. Below is a breakdown of the key points and their interpretations:

1. Accuracy (Test Accuracy): The model achieved an accuracy of 1.00 on the test set. This perfect accuracy indicates that the model correctly predicted all the test samples. It suggests that the model has generalized well to unseen data and is highly effective in classifying the test data. This is a strong performance indicator but could also imply potential overfitting, especially if the test set is not sufficiently diverse or if the dataset is small.

2. Average Absolute Error (AAE): Training Data AAE: The average absolute error on the training data is 0.0021. This low average absolute error indicates that the model's predictions are very close to the actual values for the training data. It reflects the model's accuracy in predicting the training samples, suggesting that the model fits the training data well.

3. Mean Squared Error (MSE): Test Data MSE: The mean squared error on the test set is 0.00217. The MSE measures the average squared difference between the predicted values and the actual values on the test set. A low MSE value signifies that the model's predictions are close to the actual values, reflecting good performance. However, while the MSE is low, it is essential to ensure that the model's performance is consistently good across different datasets and not just due to the test set's characteristics.

Table 1 provides a summary of the performance metrics for the Logistic Regression model, highlighting its effectiveness in classifying network traffic as either benign or malicious. The metrics include accuracy, Average Absolute Error (AAE), and Mean Squared Error (MSE) that demonstrate the model's capability in distinguishing between the two classes.

The Logistic Regression model demonstrates excellent performance with perfect accuracy on the test set, very low average absolute error on the training data, and a low mean squared error on the test set. These results indicate that the model is highly effective in classification tasks and has a good fit on both training and test data. The outlier exclusion analysis using the IQR helps to improve the quality of the dataset, ensuring that the logistic regression model can make more reliable predictions. The results indicate that the model performs exceptionally well, both in training and on unseen data, showcasing its effectiveness in classifying the target variable accurately.

Table 1 Summarizes the performance metrics of the Logistic Regression model.

Metric	Value	Interpretation
Test Accuracy	1.00	Indicates perfect classification of all test samples. Shows strong generalization.
Average Absolute Error (AAE)	0.0021	Reflects high accuracy on training data, with predictions very close to actual values.
Mean Squared Error (MSE)	0.00217	Provides an estimate of prediction error on the test set; low value suggests good performance.

4.7. CNN-GRU-LSTM Hybrid Model

The CNN-GRU-LSTM hybrid model combines Convolutional Neural Networks (CNNs), GRUs, and LSTMs to leverage the strengths of each architecture. CNNs are used to extract spatial features from data, such as patterns in network traffic data, which are then processed by GRUs and LSTMs to capture temporal dependencies [42]. This hybrid approach aims to enhance the model's ability to learn both spatial and temporal features, improving its performance in complex classification tasks like network attack detection.

The CNN-GRU-LSTM hybrid model combines the power of CNN for feature extraction, GRU for efficient sequence learning, and LSTM for handling long-term dependencies. The following is the pseudo code for CNN-GRU-LSTM Hybrid Model:

```
# CNN-GRU-LSTM Hybrid Model Pseudo Code
def cnn_gru_lstm_hybrid(input_data):
    # Step 1: Input Layer
    X = input_data  # Input data (network traffic as sequence or time series)
    # Step 2: CNN Layer for Feature Extraction
    Conv1 = Conv2D(filters=64, kernel_size=3, activation='relu')(X)
    Pool1 = MaxPooling2D(pool_size=2)(Conv1)
    # Step 3: GRU Layer for Sequence Learning
    GRU1 = GRU(units=128, return_sequences=True)(Pool1)
    GRU2 = GRU(units=128)(GRU1)
    # Step 4: LSTM Layer for Long-Term Dependencies
    LSTM1 = LSTM(units=128, return_sequences=True)(GRU2)
    LSTM2 = LSTM(units=128)(LSTM1)
    # Step 5: Fully Connected Layer
    Flattened = Flatten()(LSTM2)
    Dense1 = Dense(units=256, activation='relu')(Flattened)
    # Step 6: Output Layer
    Output = Dense(units=1, activation='sigmoid')(Dense1)  # Binary classification
    # Step 7: Compile Model
    model = Model(inputs=X, outputs=Output)
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    # Step 8: Train Model
    model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_val, y_val))
    return model
```

Table 2 presents a comprehensive overview of the CNN-GRU-LSTM model's performance, including key metrics and computational efficiency.

The CNN-GRU-LSTM model demonstrates high performance in terms of accuracy, precision, recall, and other metrics, showing it effectively classifies data and maintains a low rate of false positives and omissions. However, the high False Negative Rate suggests that it may miss a significant number of positive cases. The model's computational intensity is also reflected in its considerable test time, which might impact its practicality in real-time applications. The AUC score further validates the model's good capability in classifying between positive and negative instances (Figure 9).

Table 2 Summarizes the performance metrics and test time of the CNN-GRU-LSTM model

Metric	Value	Interpretation
Accuracy	0.998	99.8% accuracy indicates high performance in classifying test data.
Precision	0.998	99.8% precision indicates few false positives; model is reliable in predicting positives.
Recall	0.998	99.8% recall reflects the model’s effectiveness in identifying actual positives.
F1-Score	0.997	99.7% F1-score balances precision and recall well.
True Negative Rate (TNR)	0.999	99.9% TNR shows high effectiveness in predicting negatives.
Matthew's Correlation Coefficient (MCC)	0.494	Moderate MCC indicates some correlation but room for improvement.
Negative Predictive Value (NPV)	0.998	99.8% NPV suggests the model is effective in predicting true negatives.
False Discovery Rate (FDR)	0.078	7.8% FDR indicates a low rate of false positives.
False Negative Rate (FNR)	0.735	73.5% FNR shows many actual positives are missed.
False Omission Rate (FOR)	0.002	0.2% FOR is very low, indicating rare false omissions of negatives.
False Positive Rate (FPR)	0.00006	0.006% FPR shows a very low rate of false positives.
Test Time	645.40 seconds	The model requires significant time for testing, reflecting computational intensity.
AUC	0.8559	AUC of 0.8559 shows strong performance in distinguishing between classes.

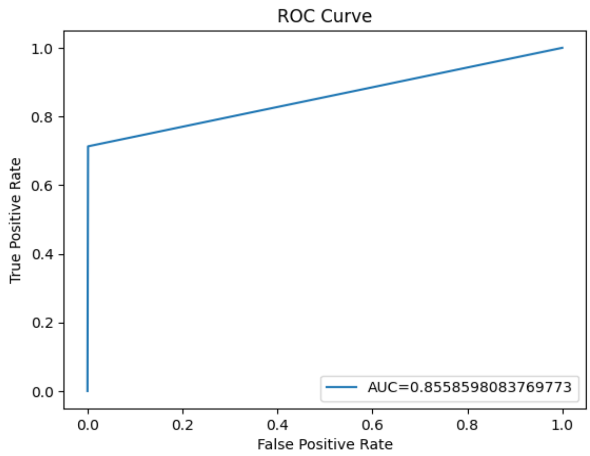


Figure 9 The area under the ROC curve (AUC).

4.8. Hyperparameters and Model Training

Hyperparameters are critical parameters that are set before the training process and influence the performance of machine learning models. For LDA, hyperparameters include the choice of the solver and regularization parameters. In Logistic Regression, hyperparameters such as the regularization strength (C) and solver type are important for controlling overfitting and optimization [43]. For neural network models, hyperparameters include the number of layers, number of units per layer, learning rate, batch size, and number of epochs.

By leveraging these models and techniques, the research aims to build a robust classification framework for detecting and categorizing network attacks, providing valuable insights into the effectiveness and efficiency of different machine learning approaches.

4.9. Advancements in CNN, GRU, and LSTM Models

In recent years, significant advancements have been made in the field of deep learning, particularly in models like Convolutional Neural Networks (CNN), Gated Recurrent Units (GRU), and Long Short-Term Memory (LSTM) networks, which have greatly enhanced their performance in complex tasks such as network attack detection. These improvements address several challenges inherent to earlier versions of these models, particularly in terms of model efficiency, generalization, and handling long-term dependencies.

Convolutional Neural Networks (CNN): Modern CNN architectures have incorporated techniques such as batch normalization, adaptive pooling, and skip connections to improve convergence and reduce overfitting. These enhancements enable CNNs to better capture spatial features in network traffic data, making them effective for feature extraction from complex network flow data. Residual networks (ResNets) and DenseNets, which introduce skip connections between layers, allow deeper CNNs to be trained more effectively, enabling more robust feature extraction for detecting malicious traffic patterns.

Gated Recurrent Units (GRU) and Long Short-Term Memory (LSTM): Recent advances in GRU and LSTM architectures have addressed the limitations of traditional recurrent neural networks (RNNs), particularly the vanishing gradient problem. New techniques, such as the peephole connections in LSTMs and the GRU with attention mechanisms, have improved their ability to capture long-term dependencies and complex temporal patterns in sequential data. Additionally, these architectures have been integrated with advanced optimization algorithms, such as Adam and RMSprop, which have significantly improved convergence rates and model stability during training. GRU and LSTM networks are now better equipped to handle the temporal nature of network traffic, which is essential for accurate attack detection over time.

These recent advancements have been incorporated into the hybrid CNN-GRU-LSTM model used in this study, enabling the model to better capture both spatial and temporal features of network traffic. By combining the strengths of CNNs for feature extraction, GRUs for temporal sequence learning, and LSTMs for long-term dependency capture, the hybrid model is particularly well-suited for the complex task of detecting network attacks.

5. Model Evaluation

Evaluating the performance of machine learning models is crucial for understanding their effectiveness in classification tasks. This involves using various metrics that measure different aspects of model performance, including accuracy, precision, recall, F1-score, and more. Here's a consolidated overview of key metrics and a comparative analysis of different models:

5.1. Metrics for Performance Evaluation

- **Accuracy**: Accuracy measures the proportion of correctly classified instances out of the total instances and is calculated as: Accuracy=Number of Correct Predictions/Total Number of Predictions. While accuracy provides a general sense of the model's performance, it can be misleading in imbalanced datasets [44].

- **Precision**: Precision, or positive predictive value, indicates the proportion of true positives out of all predicted positives: Precision=True PositivesTrue/ (True Positives + False Positives). This metric is crucial when the cost of false positives is high, such as in detecting network attacks [45].
- **Recall**: Recall, or sensitivity, measures the proportion of true positives out of all actual positives: Recall=True Positives/ (True Positives+False Negatives).
- **Recall** is particularly important in scenarios where failing to detect a positive instance is critical (Powers, 2011).
- **F1-Score**: The F1-score is the harmonic mean of precision and recall, calculated as:
- **F1-Score**= 2 *((Precision*Recall)/ (Precision + Recall)). It is especially useful for imbalanced datasets, as it accounts for both false positives and false negatives [40].
- **ROC Curve and AUC**: The Receiver Operating Characteristic (ROC) curve plots the true positive rate (recall) against the false positive rate at various thresholds. The Area Under the Curve (AUC) provides an aggregate measure of performance across all classification thresholds, with 1 indicating perfect classification and 0.5 indicating no discriminative power [47].

5.2. Comparative Analysis of Model Performance

To assess the performance of various models, we compare Linear Discriminant Analysis (LDA), Logistic Regression, and a hybrid model combining Convolutional Neural Networks (CNN), Gated Recurrent Units (GRU), and Long Short-Term Memory networks (LSTM). The comparison focuses on several critical aspects:

1 Accuracy: Reflects how well the model predicts both classes.

2 Precision, Recall, and F1-Score: Provide insights into the model's performance on each class, particularly in imbalanced datasets.

3 Error Metrics: Include Mean Squared Error (MSE) to gauge prediction accuracy and model reliability.

4 Test Time: Measures the time required for the model to make predictions on the test set, indicating computational efficiency.

In this section, we provide a detailed interpretation of the performance metrics for three different models: Linear Discriminant Analysis (LDA), Logistic Regression, and the CNN-GRU-LSTM hybrid model. Table 3 summarizes the performance metrics and test time for each model, highlighting their strengths and weaknesses in classification tasks. Each model has its strengths and weaknesses, which are critical to understanding their effectiveness in classification tasks. By examining these models closely, we can identify their capabilities and limitations in handling various types of data, particularly in the context of detecting malicious versus non-malicious instances. Below, we outline the strengths and weaknesses of each model to facilitate a comprehensive understanding of their performance.

- **LDA Model:**
  - Strengths: Achieves perfect precision, recall, and F1-scores for the "not malicious" class, indicating exceptional performance on this majority class.
  - Weaknesses: Performs poorly on the "malicious" class, with a low recall of 0.24, suggesting difficulties in identifying malicious instances due to class imbalance.
  - Overall: Highly accurate but skewed towards the majority class, indicating a need for improvement in handling the minority class.
- **Logistic Regression Model:**
  - Strengths: Achieves perfect accuracy, precision, and recall across both classes, demonstrating robustness in classification with minimal deviation.
  - Weaknesses: No significant weaknesses are noted, as it performs well on both classes.
  - Overall: Provides reliable and accurate classification.
- **CNN-GRU-LSTM Model:**
  - Strengths: Exhibits high accuracy, precision, and recall, with an impressive F1-score, effectively distinguishing between classes as indicated by a high AUC value.
  - Weaknesses: High False Negative Rate (FNR), indicating that it misses a significant number of positive instances, which can impact its utility in critical detection scenarios.
  - Test Time: Significant computational complexity is reflected in its longer test time (645.40 seconds).
  - Overall: Strong in classification performance but comes with trade-offs in computational time and higher False Negative Rate.

Table 3 Summary of Model Performance and Test Time

Metric	LDA Model	Logistic Regression Model	CNN-GRU-LSTM Model
Accuracy	1.00	1.00	0.998
Precision (Not Malicious)	1.00	1.00	0.998
Recall (Not Malicious)	1.00	1.00	0.998
F1-Score (Not Malicious)	1.00	1.00	0.997
Precision (Malicious)	0.90	1.00	0.998
Recall (Malicious)	0.24	1.00	0.998
F1-Score (Malicious)	0.38	1.00	0.997
Macro-Average Precision	0.95	1.00	0.998
Macro-Average Recall	0.62	1.00	0.998
Macro-Average F1-Score	0.58	1.00	0.997
Weighted-Average Precision	0.98	1.00	0.998
Weighted-Average Recall	0.75	1.00	0.998
Weighted-Average F1-Score	0.77	1.00	0.997
True Negative Rate (TNR)	N/A	N/A	0.999
Matthew's Correlation Coefficient (MCC)	N/A	N/A	0.494
Negative Predictive Value (NPV)	N/A	N/A	0.998
False Discovery Rate (FDR)	N/A	N/A	0.078
False Negative Rate (FNR)	N/A	N/A	0.735
False Omission Rate (FOR)	N/A	N/A	0.002
False Positive Rate (FPR)	N/A	N/A	0.00006
AUC	N/A	N/A	0.8559
Test Time	N/A	N/A	645.40 seconds

In conclusion, Linear Discriminant Analysis (LDA) and Logistic Regression are simpler models that achieve high accuracy but face challenges related to class imbalance, particularly concerning the "malicious" class. In contrast, the CNN-GRU-LSTM model offers a more balanced performance with high precision and recall for both classes; however, it requires longer training and testing times due to its complexity. When selecting a model, it is essential to consider the trade-offs between performance and computational resources. While the CNN-GRU-LSTM model is preferable for scenarios that demand high accuracy and effective handling of class imbalance, LDA or Logistic Regression may be sufficient for situations prioritizing computational efficiency. This comprehensive overview of model performance metrics supports informed decision-making in choosing the most suitable model for specific classification tasks.

In addition to describing the improvements in the CNN, GRU, and LSTM models, we have expanded the comparative analysis between our hybrid CNN-GRU-LSTM model and other state-of-the-art models used for network attack detection.

Our approach has been compared with traditional machine learning models such as Logistic Regression and Linear Discriminant Analysis (LDA), as well as with other deep learning models, including Fully Connected Networks and Single-Model CNN or LSTM architectures. This comparison not only highlights the strengths of our hybrid model but also demonstrates how it outperforms simpler models in handling complex, imbalanced datasets with both spatial and temporal dependencies.

- Logistic Regression and Linear Discriminant Analysis are simpler models that offer high accuracy in certain contexts, but they are less effective in capturing the intricate, nonlinear relationships found in complex data like network traffic. These models struggle with class imbalance, which is common in attack detection, and cannot model sequential dependencies in the data.
- Fully Connected Networks (FNNs) and Single-Model CNN/LSTM architectures are effective for certain types of data but fall short in handling both spatial features (as seen in CNN) and temporal dependencies (as seen in GRU/LSTM). While CNNs excel at extracting features, they do not explicitly model sequential dependencies, and while LSTMs are good at modeling time-series data, they lack the capacity for complex feature extraction.

The CNN-GRU-LSTM hybrid model, by combining these techniques, has been shown to be superior in handling both spatial and temporal features simultaneously, which is crucial for network attack detection where attack patterns often span both dimensions.

6. Model Interpretability- SHAP (SHapley Additive exPlanations)

Model interpretability is crucial for understanding how machine learning models make predictions and for building trust in their outputs. SHAP (SHapley Additive exPlanations) is a powerful framework for interpreting complex machine learning models. SHAP values are based on Shapley values from cooperative game theory, which provide a unified measure of feature importance by fairly distributing the prediction among all input features [48].

SHAP values decompose a model's prediction into contributions from each feature, offering a clear explanation of how each feature impacts the final prediction. This approach allows for consistent interpretation across different models, whether they are tree-based methods, neural networks, or any other complex algorithms. By assigning a Shapley value to each feature, SHAP helps identify which features are driving model decisions and to what extent. Recent advancements have extended SHAP to handle high-dimensional and complex data efficiently, making it a valuable tool in various domains, including network security [48].

6.1. Application of SHAP for Feature Importance Analysis

In the context of network attack detection, SHAP can be applied to analyze feature importance and understand model behavior. By calculating SHAP values for individual predictions, it is possible to determine which features are most influential in classifying a particular network traffic instance as benign or malicious. For example, in a model trained on network traffic data, SHAP can reveal which features, such as packet size, source IP address, or protocol type, contribute most significantly to the prediction of an attack or a benign activity.

The process involves the following steps:

Model Training: Train a machine learning model using network traffic data. This model can be a neural network, tree-based model, or any other suitable algorithm. The deep model is built and trained using Keras, where the model consists of an input layer, a hidden layer with a ReLU activation function, and an output layer with a sigmoid activation function. The model is assembled using the specified loss parameter and evaluation criteria. Figure 10 shows the progress of training the model over 10 episodes (epochs) and the evaluation on the test data after each episode. Each line in the output contains the following information: - `Epoch n/m`: where `n` shows the current episode number and `m` the total number of episodes. After training is completed, the model is evaluated using the test data, showing the average accuracy and loss of the model on the test data.

This model performs several steps to verify and explore the data before building the deep model. These steps are: Checking shapes, Filtering Invalid Values, Distribution Check, Test on Small Sample, and Data Exploration [40].

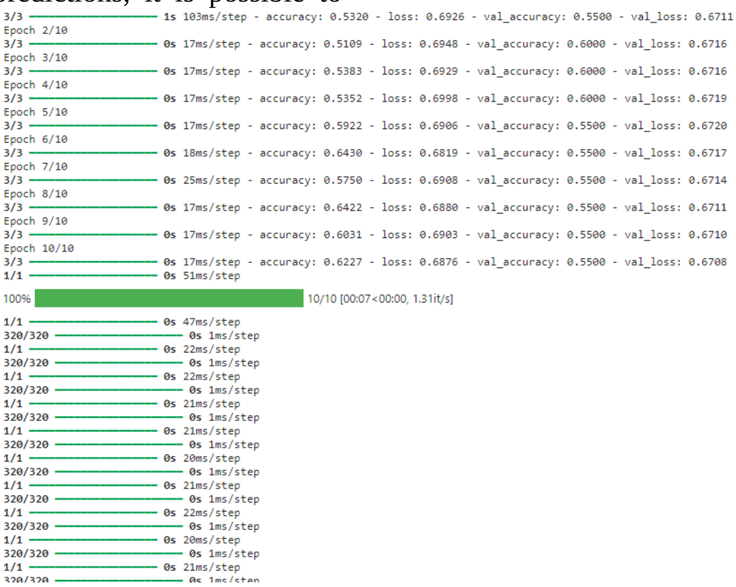


Figure 10 The progress of training the model over 10 episodes (epochs)

SHAP Value Calculation: Use the SHAP library to compute Shapley values for each feature. This involves generating a set of predictions for the input data and calculating the contribution of each feature to these predictions. Figure 11 indicates the process of calculating SHAP values for the specified instance using the SHAP parser.

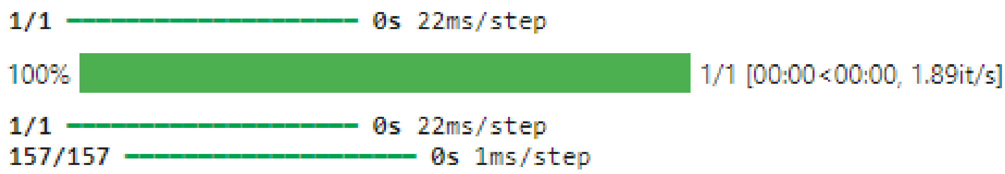


Figure 11 Output indicates the process of calculating SHAP values for the specified instance

Feature Importance Analysis: Analyze the SHAP values to identify the most influential features. This helps in understanding which aspects of the network traffic data are most predictive of different types of attacks. Figure 12 shows how Feature 1 and Feature 2 contribute to a specific prediction. If Feature 1 has a wide distribution and its SHAP value is high for a particular data point, it indicates that this feature significantly influenced the prediction for that instance.

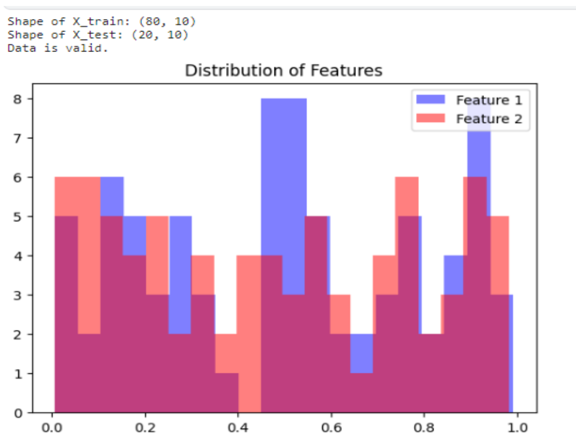


Figure 12 The distribution of two features.

By applying SHAP, practitioners can gain insights into the importance of various features and make informed decisions about feature selection and model refinement [40].

6.2. Visualization of SHAP Values

Visualizing SHAP values enhances the interpretability of machine learning models by providing intuitive and comprehensive representations of feature importance. Several types of visualizations are commonly used [40]:

Force Plots: Force plots display the contribution of each feature to a specific prediction. They show how features push the prediction score towards or away from the predicted class. For example, in network attack detection, a force plot can illustrate how different features contribute to classifying a particular network packet as malicious [49]. Figure 13 shows the contribution of each feature in determining the output.

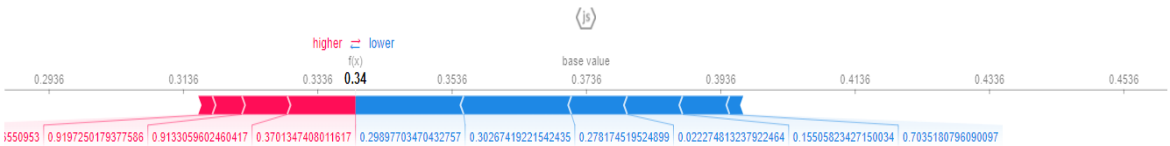


Figure 13 Force Plot SHAP values for the specified instance

Summary Plots: Summary plots aggregate SHAP values across all instances and provide a global view of feature importance. Each point on the plot represents the SHAP value of a feature for an individual instance, colored by the feature value. This visualization helps identify which features have the most significant impact across the entire dataset [49].

Figure 14 displays a simplified plot of the SHAP values calculated for each feature in the dataset, using 90 instances (10 to 100) and the specified number of samples (500). This helps in understanding the impact of each feature on the predictions made by the model.

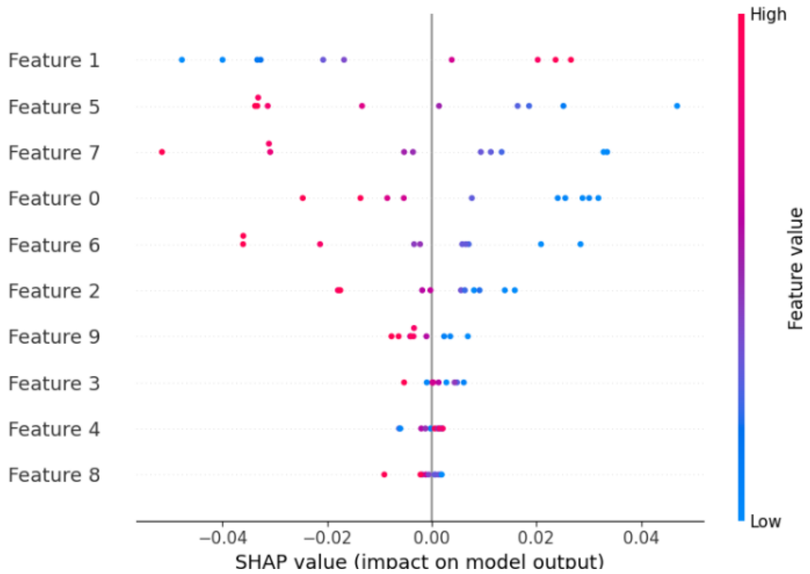


Figure 14 Summary Plots aggregate SHAP values across all instances and provide a global view of feature importance

Figure 14 provides insights into feature importance and direction. Features located farther from the center of the plot have a stronger influence on the model's predictions, with features 1, 5, and 7 exhibiting notable impact. The direction of a feature's influence is indicated by the sign of its SHAP value: positive values suggest an increase in the prediction, while negative values imply a decrease. While not explicitly shown, the plot also hints at potential feature interactions, which can be further explored by analyzing combinations of features and their corresponding SHAP values.

By employing these visualizations, stakeholders can better understand model behavior, validate the model's decisions, and gain actionable insights into the factors driving network attack detection.

7. Results and Discussion

This section details the performance metrics of three models: Linear Discriminant Analysis (LDA), Logistic Regression, and the CNN-GRU-LSTM hybrid model, summarized in Table 3. Each model exhibits distinct strengths and weaknesses in their ability to classify network traffic as either benign or malicious.

1. Linear Discriminant Analysis (LDA): LDA achieved an impressive accuracy of 99.8% on the test data. This high accuracy indicates that LDA effectively differentiates between not malicious and malicious network traffic. However, LDA's performance might be limited in handling non-linear relationships and complex attack patterns due to its reliance on linear decision boundaries.
2. Logistic Regression: The Logistic Regression model also performed exceptionally well, with an accuracy of 100% on the test set. This indicates that the model correctly classified all test instances. The model's high accuracy and low mean squared error suggest it is effective for the current dataset. However, logistic regression might not capture complex relationships in the data as well as more sophisticated models.
3. CNN-GRU-LSTM model: Exhibited high accuracy (0.998) and precision (0.998), with impressive recall (0.998) for the "malicious" class. However, it showed a high false negative rate (0.735), indicating that it misses a significant number of positive instances. Despite these limitations, the model achieved a strong AUC of 0.8559, reflecting its effectiveness in distinguishing between benign and malicious traffic. The computational complexity of this model is evident in its longer test time of 645.40 seconds.
4. Error Metrics: The models generally exhibited low mean squared errors and high F1-scores, reflecting their ability to balance precision and recall effectively. Notably, the CNN-GRU-LSTM model showed a well-rounded performance with high accuracy, precision, recall, and F1-score, but also revealed areas for improvement, such as the False Negative Rate (FNR) and False Discovery Rate (FDR).

In summary, while the LDA and Logistic Regression models demonstrate high accuracy, they struggle with class imbalance, particularly concerning malicious instances. In contrast, the CNN-GRU-LSTM model balances performance with complexity, making it preferable for scenarios demanding high accuracy, despite its longer processing times.

5. 8. Conclusions and Future Work

In conclusion, this research contributes valuable insights into the application of machine learning models for network attack detection and sets the stage for future advancements in the field. The recommendations and potential improvements outlined provide a roadmap for enhancing the effectiveness, efficiency, and interpretability of network security solutions.

This study presents a robust framework for network attack detection by integrating advanced machine learning techniques and model interpretability tools. The hybrid CNN-GRU-LSTM model demonstrated strong performance with an accuracy of 0.998 and effectively capturing complex patterns in network traffic data. The utilization of the SYN DoS dataset from the Kitsune Network Attack Dataset provided a solid foundation for our analysis and model training, ensuring that our findings are grounded in realistic and relevant data. Furthermore, the application of SHAP values not only enhanced the interpretability of the model but also offered valuable insights into feature importance, thereby building trust in the model's predictions.

Looking ahead, future work can explore several avenues to further improve network attack detection systems. One potential direction is the incorporation of additional datasets to enhance the model's generalizability and robustness against diverse attack vectors. Moreover, experimenting with ensemble methods that combine the strengths of multiple algorithms could yield even better performance. Another important aspect is the exploration of real-time detection capabilities, enabling the model to operate in live network environments where prompt response to threats is crucial.

Additionally, ongoing research into explainable AI (XAI) techniques can enhance our understanding of model decisions, allowing practitioners to interpret complex interactions between features more effectively. Lastly, developing user-friendly visualization tools for SHAP outputs could assist security analysts in quickly identifying critical indicators of attacks, facilitating faster decision-making processes. Through these future endeavors, we aim to advance the field of network security and contribute to creating safer digital environments.

References

[1]. Barry, B., Chan, H. A. Barry, B., Chan, H. (2010), Intrusion Detection Systems, In: Stavroulakis, P., Stamp, M. (eds): Handbook of Information and Communication Security pp193-205, SpringerLink. DOI:10.1007/978-3-642-04117-4_10.

[2]. Ashiku L. and Dagli C.H. (2021). Network Intrusion Detection System using Deep Learning, Procedia Computer Science 2021, 185(1):239-247

[3]. Gottapu S. R. and Krishna S. P. (2023), A Novel Approach for Detection of DoS / DDoS Attack in Network Environment using Ensemble Machine Learning Model. International Journal on Recent and Innovation Trends in Computing and Communication 11(9):244-253. DOI: 10.17762/ijritcc.v11i9.8340ISBN: 2321-8169

[4]. Gottapu S. R. and Krishna S. P. (2024), Exploring a novel framework for DoS/DDoS attack detection and simulation in contemporary networks, January 2024i-manager’s Journal on Software Engineering 18(3):43. DOI:10.26634/jse.18.3.20596

[5]. Inuwa, M. M., & Das, R. (2024). A comparative analysis of various machine learning methods for anomaly detection in cyber-attacks on IoT networks. Internet of Things, 26, 101162. https://doi.org/10.1016/j.iot.2024.101162

[6]. Becerra-Suarez, F.L., Tuesta-Monteza, V.A., Mejia-Cabrera, H.I., Arcila-Diaz, J. (2024). Performance Evaluation of Deep Learning Models for Classifying Cybersecurity Attacks in IoT Networks. Informatics 2024, 11, 32. https://doi.org/10.3390/informatics11020032

[7]. Liu, H.; Lang, B. Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. Appl. Sci. 2019, 9, 4396. https://doi.org/10.3390/app9204396

[8]. Amutha S., Kavitha R., Srinivasan R. and Kavitha M., "Secure network intrusion detection system using NID-RNN based Deep Learning," 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2022, pp. 1-5, doi: 10.1109/ACCAI53970.2022.9752526.

[9]. Liao, H., Murah, M. Z., Hasan, M. K., Aman, A. H. M., Fang, J., Hu, X., & Khan, A. U. R. (2024). A Survey of Deep Learning Technologies for Intrusion Detection in Internet of Things. IEEE Access. vol.12, pp.4745-4761, 2024.

[10]. Tossou, S., Qorib, M., & Kacem, T. (2023, October). Anomaly Based Intrusion Detection System:

A Deep Learning Approach. In 2023 International Symposium on Networks, Computers and Communications (ISNCC) (pp. 1-6). IEEE. Doha, Qatar, 2023, pp. 1-6, doi: 10.1109/ISNCC58260.2023.10323740.

[11]. Rani, S., & Kumar, S. (2023, May). Unleashing the Power of Machine and Deep Learning for Advanced Network Intrusion Detection: An Analysis and Exploration. In 2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI) IEEE, Chennai, India, 2023, pp. 1-9, doi: 10.1109/ACCAI58221.2023.10200892.

[12]. Pandathara A. (2023). A Comprehensive Examination of Literature Exploring the Implementation of Machine Learning to Network Security's Intrusion Detection Systems. International Journal of Advanced Research in Science, Communication and Technology, doi: 10.48175/ijarsct-8605

[13]. Hussain, A., Sharif, H., Rehman, F., Kirn, H., Sadiq, A., Khan, M. S., Riaz, A., Ali, C. N., & Chandio, A. H. (2023). A systematic review of intrusion detection systems in internet of things using ML and DL. In 2023 4th International Conference on Computing, Mathematics and Engineering Technologies (iCoMET) (pp. 1-5). IEEE. https://doi.org/10.1109/iCoMET57998.2023.10099142

[14]. Nakip M., Gül B. C., Gelenbe E. (2023). Decentralized Online Federated G-Network Learning for Lightweight Intrusion Detection. In 2023 31st International Symposiumon Modeling, Analysis, and Simulation of Computer andTelecommunication Systems (MASCOTS). IEEE, 2023, pp. 1–8. DOI:10.1109/MASCOTS59514.2023.10387644

[15]. Zhu, S., Xu, X., Zhao, J., & Xiao, F. (2024). Lkd-stnn: A lightweight malicious traffic detection method for internet of things based on knowledge distillation. IEEE Internet of Things Journal, vol. 11, no. 4, pp. 6438-6453, 15 Feb.15, 2024.

[16]. Kheddar, H., Himeur, Y., & Awad, A. I. (2023). Deep transfer learning for intrusion detection in industrial control networks: A comprehensive review. Journal of Network and Computer Applications, 220, 103760. https://doi.org/10.1016/j.jnca.2023.103760

[17]. Sunil, C. K., Reddy, S., Kanber, S. G., Sandeep, V. R., & Patil, N. (2023). Comparative analysis of intrusion detection system using ML and DL techniques. In Hybrid Intelligent Systems (pp. 736-745). Springer, Cham. https://doi.org/10.1007/978-3-031-27409-1_67

[18]. Mert, Nakip., Baran, Can, Gül., Erol, Gelenbe. (2023). Decentralized Online Federated G-Network Learning for Lightweight Intrusion Detection. arXiv.org, doi: 10.48550/arXiv.2306.13029

[19]. Wasnik P. and Chavhan N., "A Review Paper on Designing Intelligent Intrusion Detection System Using Deep Learning," 2023 11th International Conference on Emerging Trends in Engineering & Technology - Signal and Information Processing (ICETET - SIP), Nagpur, India, 2023, pp. 1-6, doi: 10.1109/ICETET-SIP58143.2023.10151563.

[20]. Ogundokun R. O., Basil U., Babatunde A. N., Abdulahi A. T., Adenike A. R. and Adebiyi A. A., "Intrusion Detection Systems Based on Machine Learning Approaches: A Systematic Review," 2023 International Conference on Science, Engineering and Business for Sustainable Development Goals (SEB-SDG), Omu-Aran, Nigeria, 2023, pp. 01-04, doi: 10.1109/SEB-SDG57117.2023.10124506.

[21]. Krishna, A., Lal, A., Mathewkutty, A. J., Jacob, D. S., & Hari, M. (2020, July). Intrusion detection and prevention system using deep learning. In 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC) (pp. 273-278). IEEE.

[22]. Fadel, M. M., El-Ghamrawy, S. M., Ali-Eldin, A. M., Hassan, M. K., & El-Desoky, A. I. (2022). HDLIDP: A Hybrid Deep Learning Intrusion Detection and Prevention Framework. Computers, Materials & Continua, 73(2).

[23]. Alghamdi, Mohammed I., A Hybrid Model for Intrusion Detection in IoT Applications, Wireless Communications and Mobile Computing, 2022, 4553502, 9 pages, 2022. https://doi.org/10.1155/2022/4553502

[24]. Monani A. Bhusnale O. Borade K. Madali R. (2023). Analysing Cyber Threats: A Comprehensive Literature Review on Data-Driven Approaches, International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), Volume 9, Issue 3, pp.188-193, May-June-2023. Available at doi : https://doi.org/10.32628/CSEIT2390351

[25]. Auwal, Sani, Iliyasu. (2022). A Survey of Network Intrusion Detection Techniques Using Deep Learning. International Journal of Engineering Research in Computer Science and Engineering, doi: 10.36647/ijercse/09.08.art017

[26]. [28] Hnamte V., Hussain J. (2023). Network Intrusion Detection using Deep Convolution Neural Network. 4th International Conference for Emerging Technology (INCET). doi: 10.1109/INCET57972.2023.10170202

[27]. Alabdulatif, A., & Rizvi, S.S.H. (2022). Machine learning approach for improvement in kitsune NID. Intelligent Automation & Soft Computing, 32(2), 827-840. https://doi.org/10.32604/iasc.2022.021879

[28]. Malliga, S., Nandhini, P. S., & Kogilavani, S. V. (2022). A comprehensive review of deep learning techniques for the detection of (distributed) denial of service attacks. Information Technology and Control, doi: 10.5755/j01.itc.51.1.29595

[29]. Sujatha, V., Prasanna, K. L., Niharika, K., Charishma, V., & Sai, K. B. (2023). Network intrusion detection using deep reinforcement learning. 2023 7th International Conference on Computing Methodologies and Communication (ICCMC), 1146-1150. https://doi.org/10.1109/ICCMC56507.2023.10083673

[30]. Mohammed, A., Bahashwan, A.A., Manickam, S., Al-Amiedy, T.A., Aladaileh, M.A., & Hasbullah, I.H. (2023). A systematic literature review on machine learning and deep learning approaches for detecting DDoS attacks in software-defined networking. Sensors, 23(9), 4441. https://doi.org/10.3390/s23094441

[31]. Omarov, B., Asqar, M., Sadybekov, R., Koishiyeva, T., Bazarbayeva, A., & Uxikbayev, Y. (2022). IoT network intrusion detection: A brief review. 2022 International Conference on Smart Information Systems and Technologies (SIST), 1-5. https://doi.org/10.1109/SIST54437.2022.9945763

[32]. Tahreeem, M., Andleeb, I., Hussain, B. Z., & Hameed, A. (2022, December). Machine learning-based Android intrusion detection systems. Paper presented at the International Conference on Data Intensive Applications & Their Challenges (Computatia X), Jaipur, India. Aligarh Muslim University, University of Windsor, Texas A&M University.

[33]. Gonaygunta, H. (2023). Machine learning algorithms for detection of cyber threats using logistic regression. International Journal of Smart Sensor and Adhoc Network, 3(4), 36-42. https://doi.org/10.47893/IJSSAN.2023.1229

[34]. Pandey, G., Kumar, A. K., & Jha, M. (2024). Human activity recognition using CNN-LSTM-GRU model. International Research Journal on Advanced Engineering Hub (IRJAEH), 2(04), 889-894. https://doi.org/10.47392/IRJAEH.2024.012

[35]. Bhattarai, A., Gyawali, U., Verma, A., & Ranga, V. (2024). Improving intrusion detection in a software-defined network using hybrid CNN and Bi-LSTM. Proceedings of the 2024 IEEE International Conference on Artificial Intelligence and Computational Applications (ICAAIC). https://doi.org/10.1109/icaaic60222.2024.10575090

[36]. Abdulhakim, A., & Ilyas, M. (2024). Deep learning for smart grid intrusion detection: A hybrid CNN-LSTM-based model. International Journal of Artificial Intelligence & Applications (IJAIA), 15(3), 1-10. https://doi.org/10.5121/ijaia.2024.15301

[37]. Al-Aql, N. (2024). Hybrid RNN-LSTM networks for enhanced intrusion detection in vehicle CAN systems. Journal of Electrical Systems, 33(1), 1-8. https://doi.org/10.52783/jes.3318

[38]. Poornachander, V., Kumar, K. S., & Jagadish, S. (2024). DDoS attack intrusion detection system with CNN and LSTM hybridization. Proceedings of the 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 1-6. https://doi.org/10.1109/ICSCSS60660.2024.10625330

[39]. Lv, H., & Ding, Y. (2024). A hybrid intrusion detection system with K-means and CNN+LSTM. EAI Endorsed Transactions on Scalable Information Systems, 11(6). https://doi.org/10.4108/eetsis.5667

[40]. Abu Khalil, D., & Abuzir, Y. (n.d.). Detecting and Analyzing Network Attacks: A Time-Series Analysis Using the Kitsune Dataset. Journal of Emerging Computer Technologies, 5(1), 9-23. https://doi.org/10.57020/ject.1563146.

[41]. Gür, Y. E. (2024). Comparative Analysis of Deep Learning Models for Silver Price Prediction: CNN, LSTM, GRU and Hybrid Approach. Akdeniz İİBF Dergisi, 24(1), 1-13. https://doi.org/10.25294/auiibfd.1404173

[42]. Scikit-learn. (2021). scikit-learn: Machine Learning in Python. Retrieved from https://scikit-learn.org/

[43]. Hayel, R., Hindi, K. M., Hosny, M. I., & Alharbi, R. (2024). A selective LVQ algorithm for improving instance reduction techniques and its application for text classification. Journal of Intelligent & Fuzzy Systems. https://doi.org/10.3233/JIFS-235290

[44]. Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. Proceedings of the 23rd International Conference on Machine Learning (ICML 2006).

[45]. Van Rijsbergen, C. J. (1979). Information Retrieval. Butterworth-Heinemann.

[46]. Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27(8), 861-874.

[47]. Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017), 4765-4774.

[48]. Lundberg, S. M., Erion, G., & Lee, S. I. (2020). Explainable AI for Trees: From Local Explanations to Global Understanding. Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (FAccT 2020), 418-429.

Journal of Soft Computing and Artificial Intelligence

Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access

Volume 05
Issue 02

December, 2024

Research Article

A Hybrid CNN-LSTM Model for Predicting Energy Consumption and Production Across Multiple Energy Sources

***Mahmut Kaya*[1], *Anıl Utku*[2], *Yavuz Canbay*[3]**

[1]*Department of Artificial Intelligence and Data Engineering, Faculty of Engineering, Fırat University, Elazığ, Turkey*
[2]*Department of Computer Engineering, Faculty of Engineering, Munzur University, Tunceli, Turkey*
[3]*Department of Computer Engineering, Faculty of Engineering and Architecture, Sutcu Imam University, Kahramanmaraş, Turkey*

ARTICLE INFO

Article history:
Received November 1, 2024
Revised November 18, 2024
Accepted December 3, 2024

Keywords:
Energy Prediction
Deep Learning
CNN
LSTM

ABSTRACT

This study leverages a comprehensive dataset provided by Energy Exchange Istanbul (EXIST), a prominent authority in energy data, encompassing hourly energy consumption and production data across Turkey. To enhance the accuracy of energy consumption and production forecasting, a variety of machine learning and deep learning models were employed, including linear regression (LR), random forest (RF), support vector regression (SVR), convolutional neural networks (CNN), long short-term memory networks (LSTM), and the proposed hybrid CNN-LSTM model. The study reformulates the time series data into a regression problem by applying the sliding window technique. The experimental findings reveal that the hybrid CNN-LSTM model outperforms other models in forecasting total energy consumption as well as the production of natural gas, hydro dam, lignite, hydro river, wind, and fuel oil. The hybrid model achieved superior performance metrics, including the lowest root mean square error (RMSE) and mean absolute error (MAE) values, alongside the highest coefficient of determination (R^2) scores. The enhanced predictive capability of the proposed approach is attributed to the synergistic combination of CNN's strength in capturing local patterns and LSTM's proficiency in modeling long-term temporal dependencies. This study underscores the effectiveness of the hybrid CNN-LSTM model in accurately forecasting energy consumption and production, thereby contributing significantly to the efficient utilization of energy resources and supporting informed decision-making in energy management. Experiments showed that CNN-LSTM outperforms the compared models with above 0.999 R^2.

1. Introduction

Over the last few decades, there has been an unprecedented increase in electricity demand worldwide. The main reasons for this increase include rapid population growth, significant technological advances, and the proliferation of electronic devices that have become integral to modern life. As the world's population grows, energy consumption rises parallel, necessitating a reliable and abundant electricity supply. Technological advances lead to the emergence of new energy-intensive industries and processes, further increasing demand. In light of these complexities, effective power system planning and management has become

* Corresponding author
e-mail: mahmutkaya@firat.edu.tr
DOI: 10.55195/jscai.1577431

more critical than ever to ensure that electricity systems remain reliable and manageable. Accurate energy demand forecasting is not just essential but urgent to avoid system overloads and blackouts, which can have serious economic and social consequences.

Achieving a balanced and efficient energy system requires developing and applying precise and accurate forecasting models for energy production and consumption [1, 2]. These models are vital tools that help energy providers predict future needs and adjust their operations accordingly. By accurately forecasting demand, energy companies can optimize production schedules, manage resources more effectively, and reduce operational costs.

Accurate forecasting of energy consumption also facilitates better allocation of energy resources, allowing providers to distribute energy where and when it is most needed. Furthermore, accurate forecasts provide a systematic roadmap for energy savings and energy infrastructure improvements [3]. The importance of forecasting energy demand and generation consumption is growing every day, mainly as the world tackles the challenges of climate change and sustainable development.

Accurate forecasts provide decision-makers in industry and governments with invaluable guidance and reference points on future directions and plans [4]. Based on these forecasts, policymakers design regulations and incentives that promote efficient energy use, support renewable energy adoption, and ensure energy security. Accurate forecasts inform industry investment decisions, operational strategies, and risk management practices.

Data-driven models have played an important role in accurately forecasting energy production and consumption. Traditional regression algorithms and time series analysis techniques were used for electricity forecasting. While these methods provide a basic understanding, they often lack the ability to capture the detailed patterns and complexities in energy data. Factors such as weather variations, consumer behavior, economic fluctuations, and technological changes introduce non-linearities and interactions that traditional models may not adequately address.

As a result, as global electricity demand continues to grow, the role of accurate energy forecasting models is becoming increasingly critical. Using advanced data-driven approaches enables us to improve the reliability and efficiency of power systems while supporting sustainable energy initiatives. Accurate forecasts provide valuable insights for strategic decision-making in the public and private sectors, facilitating the development of policies and practices that balance economic growth with environmental responsibility.

Today, machine learning and deep learning techniques have become essential tools for addressing the increasing volume and complexity of data [5]. In various sectors such as energy, healthcare, finance, transportation, and agriculture, these methods enable the extraction of meaningful insights from large datasets, leading to more accurate predictions and effective decision-making mechanisms. While machine learning excels in identifying patterns and performing classifications through data-driven algorithms, deep learning offers advanced solutions by capturing complex structures and modeling long-term dependencies. The widespread application of these techniques not only enhances operational efficiency but also facilitates the sustainable and efficient management of resources. Therefore, machine learning and deep learning stand out as cornerstone technologies in the ongoing digital transformation of our era.

2. Related Works

This section reviews some important studies on energy consumption and production. First of all, in [1], the authors emphasize that existing studies generally ignore user behavior in energy consumption forecasting and propose to develop a cooling energy consumption forecasting model that considers user behavior to overcome this deficiency. For this purpose, a new model was tested using four machine learning algorithms: artificial neural network (ANN), deep neural network (DNN), classification and regression tree (CART), and bagging tree. A simulation dataset consisting of 3 months of hourly data, including 5760 energy usage cases, was used, and coefficient of variation, mean square error (MSE) and R^2 metric were applied for performance evaluations. The results show that all algorithms achieve accurate prediction results when sufficient data is used.

Similarly, Dong et al. [6] presented an energy consumption forecasting model to solve this problem, noting that existing research does not consider building operating conditions over different periods. In the experiments, hourly collected meteorological data and energy consumption data of an office building are used. Using an ensemble learning method, the proposed model extracted energy consumption patterns and built an energy consumption model for each pattern. This model was compared with an ANN and support vector regression (SVR), and the results showed that it outperformed the other models in terms of MSE and coefficient of variation of mean square error (CVMSE).

Furthermore, Hora et al. [7] emphasized that energy consumption forecasting is a challenging problem, and its importance is increasing day by day. They pointed out the need for a reliable and accurate forecasting model. Accordingly, they introduced a new meta-heuristic-based LSTM network model for energy consumption prediction. Using two publicly available datasets, they compared their method with LR, CNN, SVR, LSTM, and bidirectional long-short-term memory (BiLSTM) algorithms. They used the metrics of mean absolute percentage error (MAPE), MAE, MSE for performance evaluation. The results showed that the proposed model provided lower error rates.

Conversely, Wang et al. [3] proposed a novel methodology for consumption forecasting based on a LSTM algorithm. The performance of the proposed model is compared with that of the autoregressive moving average model (ARMA), the autoregressive fractionally integrated moving average model (ARFIMA), and the back propagation NN algorithms. The metrics of mean squared error (MSE), MAE, and MAPE are used for the comparison. Experiments with a dataset collected from a real industrial system demonstrated that the proposed model outperformed the other models.

In addition, Nie et al. [8] proposed a new hybrid method for home energy consumption based on a gradient-boosting regression tree (GBRT) and an autoregressive fractionally integrated moving average model. The proposed model is compared with iterative NN, SVR, ARFIMA model with a GBRT, and an ARFIMA model with an iterative neural network. A simulation dataset generated using the Simulink program is used. The experimental results show that the proposed hybrid method offers lower error rates regarding mean absolute, percentage, and squared errors.

In [9], highlighting the importance of solar power generation forecasting, the authors introduce a hybrid model combining a CNN, LSTM and a transformer. In the experiments, a publicly available dataset called Fingrid is used. The metrics of MAE, MAPE and MSE were used for performance evaluation. The experiments show that the proposed model offers lower error rates than six other methods: AB-net, GRU-CNN, ARIMA, DeepAR, and Prophet.

In [10], it is stated that fossil fuel power plants are harmful to human life, and more environmentally friendly solutions should be used for energy production. In this context, the authors focused on wind power generation forecasting. They used Extreme Gradient Boosting (XGBoost), Bayesian optimized multilayer perceptron, GBRT, ensemble method (gradient boosting and XGBoost), CNN, and LSTM hybrid model. In the experiments, the metrics of MAE, MAPE, MSE, and MSE were applied, and the results showed that the Bayesian optimized multilayer perceptron offers minimum error rates with acceptable runtime.

Alaraj et al. [11] emphasized that electrical energy consumption is increasing daily and that using renewable energy sources has become mandatory to meet the required consumption. This has made it even more important to predict the energy production of solar photovoltaic power plants. Therefore, the authors proposed a model based on the community tree approach and performed experimental studies on a collected dataset. The metrics of MSE, MAE, MAPE are used as performance indicators, and the results show that the proposed model offers lower error rates.

Similarly, in reference [12], the authors concentrated on a comparative analysis of the performance of machine learning algorithms for solar power generation forecasting. A variety of algorithms were employed, including SVM, k-nearest neighbor, random forest, artificial neural networks, naive Bayes, logistic regression, decision tree, gradient boosting, adaptive boosting and stochastic gradient descent. The experiments were conducted using a collected dataset, with the area under the curve and MAPE applied as performance metrics. The results demonstrated that SVM yielded the most optimal outcomes.

Finally, Ledmaoui et al. [13] presented a comparative study for solar power generation forecasting. In this study, Extreme Gradient Boosting (XGBoost), Gradient Boosting Machine (GBM), recurrent neural network (RNN), and ANN are compared. A collected dataset was used for the experiments, and MSE, MAE, and R^2 metric were applied for performance evaluation. As a result, it is observed that ANN gives the best prediction results compared to other methods.

Overall, these studies demonstrate the effectiveness of different approaches and models in energy consumption and production forecasting. Machine learning and artificial intelligence-based methods seem to provide higher accuracy and efficiency in energy forecasting. Therefore, developing these methods further and applying them to different energy sources in future research will be important.

3. Material and Methods

3.1 Dataset

This study used a dataset consisting of energy consumption and production values provided by Energy Exchange Istanbul (EXIST). The dataset used publicly available via [14]. The dataset includes hourly energy consumption and production values in Megawatts between 01/01/2018 and 31/12/2023 throughout Turkey. The dataset used consists of the attributes time, total production and consumption, natural gas, hydro dam, lignite, hydro river, coal imported, wind, solar, fuel oil, geothermal, asphalted coal, hard coal, biomass, naphtha, LNG, international, waste heat, TRY/MWh, USD/MWh and EUR/MWh. This study selected the attributes of total consumption amount and natural gas, hydro dam, lignite, hydro river, wind, and fuel oil production amounts. Figure 1 shows the change graphs of the values of the selected attributes over time.

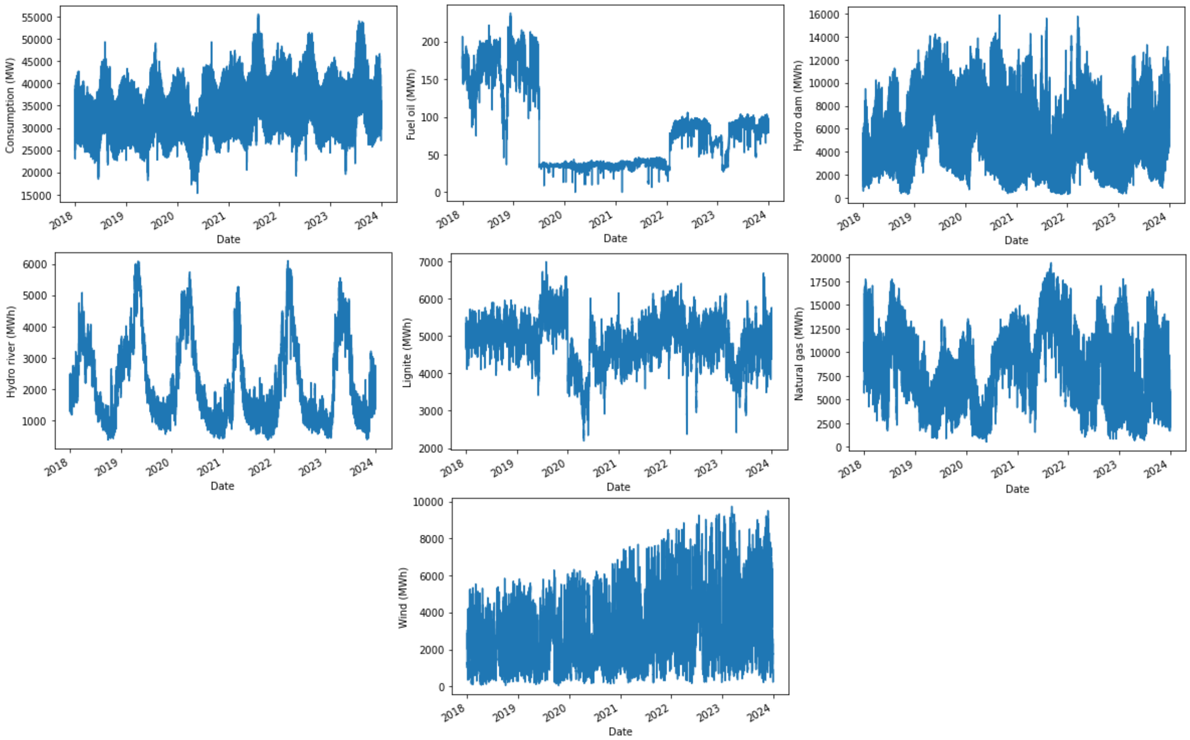


Figure 1. The change graphs of the values of the selected attributes over time

In Figure 1, sharp increases and decreases in consumption or production are observed according to seasons and periods, economic changes, and energy demand. The change in total production and total consumption over time is shown in Figure 2.

Figure 2 shows the change in total energy production and total consumption over time. Consumption and production values are generally close to each other, but in some periods, production is greater than consumption. The change in total consumption by month is shown in Figure 3.

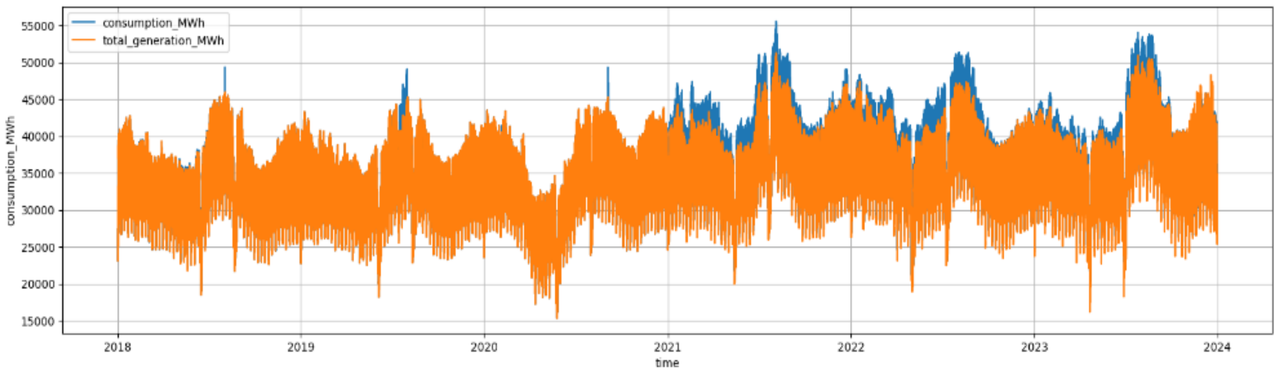


Figure 2. The change in total production and total consumption over time

Figure 3 shows the change in energy consumption by month. While the lowest energy consumption is observed in April, the highest energy consumption is observed in August. There are differences between the other months according to seasonal changes.

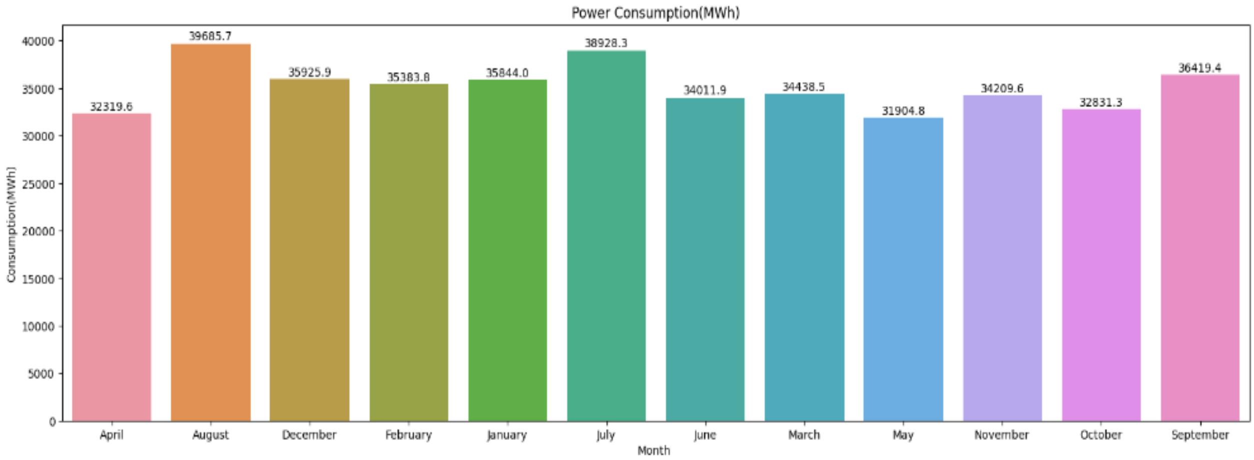


Figure 3. The change in total consumption by months

3.2 Prediction Models

In this study, a dataset provided by Energy Exchange Istanbul (EXIST) was utilized, containing hourly energy consumption and production values across Turkey between January 1, 2018, and December 31, 2023. To predict energy consumption and production more accurately and reliably across different energy sources, various machine learning and deep learning methods were employed, including LR, RF, SVR, CNN, and LSTM. Linear regression was chosen for its simplicity and interpretability, while methods like random forest and SVR excel in capturing complex data relationships. Deep learning models such as CNN and LSTM are particularly effective in learning hidden patterns and temporal dependencies in time-series data. The selected features from the dataset include time, total consumption amount, and production amounts of natural gas, hydro dam, lignite, hydro river, wind, and fuel oil. This comprehensive approach aims to achieve more precise predictions of energy consumption and production, thereby promoting more efficient use of energy resources.

The technique of linear regression [15] represents a fundamental statistical methodology which models the linear relationship between a dependent variable and one or more independent variables. This is achieved by fitting a linear equation to observed data. The simplicity and interpretability of linear regression make it an advantageous technique for regression tasks, allowing for quick implementation and computational efficiency. As a robust baseline model for predictive analysis, linear regression provides clear insights into how each predictor affects the outcome, thus helping to identify critical factors influencing energy consumption and production. Furthermore, its transparent nature makes it valuable for understanding the direct impact of individual energy sources on overall energy metrics [16].

Random Forest [17] is an ensemble learning method that constructs multiple decision trees with randomness in the tree-building process—selecting random subsets of features and samples—to reduce overfitting and improve model generalization by averaging their predictions. This approach enhances robustness to noise and outliers, captures complex nonlinear relationships without extensive parameter tuning, and effectively handles large, high-dimensional datasets by modeling interactions between variables. In the context of energy forecasting, Random Forest discerns intricate patterns in consumption and production data across different energy sources, leading to more accurate and reliable predictions [18, 19].

Support Vector Machines [20] are supervised learning models that are employed for the purposes of classification and regression.; in regression tasks, known as SVR [21], they aim to find a function that deviates from the observed values by no more than a specified margin, employing kernel functions to transform input data into a higher-dimensional space where a linear relationship can be established. SVRs offer a number of advantages in modelling non-linear relationships using a variety of kernel functions. They are particularly effective in high-dimensional spaces and demonstrate resilience to overfitting, especially when the number of dimensions exceeds the number of samples. In energy consumption and production prediction, SVRs can capture subtle and complex dependencies between different energy sources and consumption patterns [22].

CNNs are deep learning models that process grid-like data by learning spatial hierarchies of features through convolutional layers and backpropagation, utilizing components like convolution, pooling, and fully connected layers [23]. Adapted for regression tasks with time-series data, CNNs capture local patterns and trends by treating temporal data similarly to spatial data, requiring fewer parameters than fully connected networks and reducing the risk of overfitting [24]. In energy forecasting, CNNs effectively learn temporal and spatial correlations in consumption and production data across different energy sources, enhancing the accuracy of predictions [25].

Long Short-Term Memory (LSTM) networks represent a specific type of recurrent neural network, designed for the purpose of modelling sequential data. This is achieved through the utilisation of memory cells, which maintain information over extended periods, effectively addressing the vanishing gradient problem that is inherent to traditional RNNs. Particularly beneficial for time-series forecasting in regression tasks, LSTMs excel at learning temporal patterns and trends from historical data [26-28]. In the context of energy consumption and production prediction, they can model the dynamic behavior of energy systems by capturing seasonal variations and temporal dependencies, handling sequences of varying lengths, and focusing on relevant time steps, making them highly effective for accurate and reliable energy forecasting [29].

3.3 Created Hybrid Prediction Model

In this study, a time series dataset of hourly energy consumption and production amounts was used. To process time series data with machine learning or deep learning, datasets must be transformed into regression problems. The sliding window method is used to transform time series data into a regression problem structure. The sliding window method allows the data to be structured as input/output according to the specified window size, as seen in Figure 4. As seen in Figure 1, in a scenario where the window size is 3, t_1, t_2 and, t_3 will be configured as input, and t_4 will be configured as output. The sliding window progresses by shifting one observation data to the right at each step.

A series of experimental studies were conducted with the objective of determining the optimal sliding window size. The models had the lowest error rate when the sliding window size was 3. The data was input/output structured using sliding windows and scaled using MinMaxScaler.

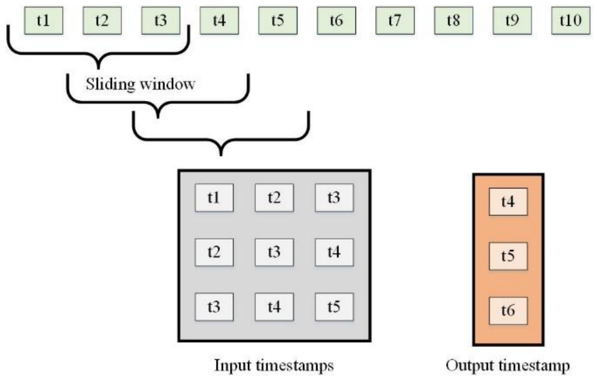


Figure 4. Sliding window method

80% of the data set was used for model training and 10% of the training data was used as validation data for model hyper-parameter determination. The models were tested on 20% of the dataset. Grid search was used to determine the hyper-parameters of the models. Grid Search is a common method used to perform hyper-parameter optimization in machine learning and deep learning models. In this process, all possible combinations for a given set of hyper-parameters are tested and the performance of the model is evaluated.

The hyper-parameter combination that provides the best performance is selected. The structure of the created CNN-LSTM model is presented in Figure 5.

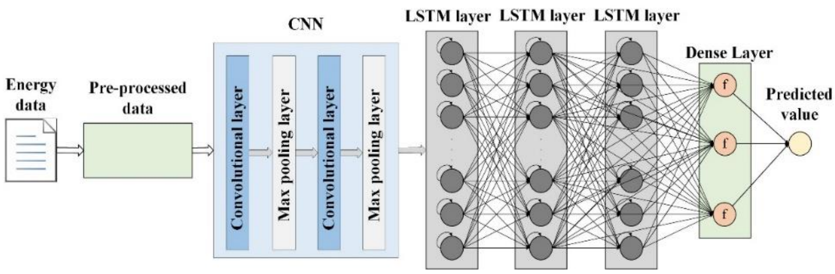


Figure 5. The structure of the created CNN-LSTM model

As seen in Figure 5, the created CNN-LSTM model consists of CNN and LSTM components. CNN was used for feature extraction and determining patterns between data. LSTM was used to learn long and short term dependencies between data. The created CNN-LSTM model takes hourly energy production and consumption values as input. The data is presented as input to CNN after being pre-processed using sliding window and MinMax scaler. CNN component uses convolution and pooling layers to extract features and determine patterns between data. LSTM component processes the feature maps provided by CNN component and enables modeling the relationships between data and learning long-short term dependencies. For the CNN component determined using grid search, the number of convolution and pooling layers is 2; the activation function is ReLU, the pool size is 2 and the number of filters is 32. LSTM component has 32 neurons and Adam optimizer is used. Dropout rate is 0.2, epoch is 80 and batch size is 8.

Hyper-parameters for LR are fit_intercept: True and normalize: False. Hyper-parameters for RF are 100, n_estimators: 100, max_depth: 20, min_samples_split: 2, min_samples_leaf: 1, and max_features: 'auto'. Hyper-parameters for SVR are C:10, kernel: linear, gamma=1e-07, and epsilon=0.1. Hyper-parameters for CNN: number of filters: 64, kernel_size: (3, 3), pool_size: (2, 2), activation function: relu, optimizer: adam, batch_size: 64, and epochs: 100. Hyper-parameters for LSTM: number of neurons: 100, dropout: 0.2, recurrent_dropout: 0.2, batch_size: 64, learning_rate: 0.001, optimizer: adam, and epochs: 100.

4. Experimental Results

In this study, a hybrid CNN-LSTM model was created for the prediction of total consumed energy and production of natural gas, hydro dam, lignite, hydro river, wind, and fuel oil. The created model was comprehensively compared with LR, RF, SVR, CNN and LSTM. Table 1 shows the experimental results for predicting total energy consumption.

Table 1. The experimental results for predicting total energy consumption

Model	RMSE	MAE	R^2
LR	1086.87	761.63	0.964
RF	1073.88	755.95	0.965
SVR	1034.90	747.93	0.968
CNN	927.85	639.22	0.974
LSTM	834.40	597.13	0.981
CNN-LSTM	741.27	541.07	0.990

In Table 1, it is observed that all models are generally successful in predicting total energy consumption. The compared models had R^2 values above 0.9. CNN-LSTM was more successful than the compared models with the lowest MAE and RMSE and the highest R^2. Table 2 presents the experimental results for predicting natural gas production.

Table 2. The experimental results for predicting natural gas production

Model	RMSE	MAE	R^2
LR	740.01	513.13	0.965
RF	732.59	501.57	0.965
SVR	731.35	498.86	0.966
CNN	729.71	498.23	0.966
LSTM	686.92	431.74	0.972
CNN-LSTM	524.48	357.27	0.984

Table 2 shows that LSTM and CNN-LSTM are more successful than other models. LR, RF, SVR and CNN had close results. CNN-LSTM was more successful than the models compared with an R2 of 0.984. Table 3 presents the experimental results for predicting fuel oil production.

Table 3. The experimental results for predicting fuel oil production

Model	RMSE	MAE	R^2
LR	3.07	1.66	0.969
RF	3.00	1.58	0.971
SVR	2.99	1.52	0.971
CNN	2.86	1.36	0.973
LSTM	2.75	1.09	0.980
CNN-LSTM	1.13	0.63	9.992

Table 3 shows that LR, RF, SVR and CNN have close results. CNN-LSTM outperformed the compared models with 1.13 RMSE, 0.63 MAE, and 0.992 R^2. Table 4 presents the experimental results for predicting hydro river production.

Table 4. The experimental results for predicting hydro river production

Model	RMSE	MAE	R^2
LR	111.86	86.32	0.989
RF	109.73	83.85	0.993
SVR	105.30	79.76	0.993
CNN	105.13	79.60	0.993
LSTM	101.84	74.47	0.995
CNN-LSTM	92.04	61.79	0.998

As seen in Table 4, all models except LR had R2 above 0.9. CNN-LSTM outperformed the compared models with 92.04 RMSE, 61.79 MAE, and 0.998 R^2. Table 5 presents the experimental results for predicting hydro dam production.

Table 5. The experimental results for predicting hydro dam production

Model	RMSE	MAE	R^2
LR	698.38	535.45	0.908
RF	647.10	497.02	0.921
SVR	632.55	484.06	0.925
CNN	622.91	483.62	0.927
LSTM	573.80	419.22	0.943
CNN-LSTM	489.93	333.07	0.961

As seen in Table 5, LR is relatively less successful than other models. SVR and CNN gave similar results and were more successful than RF. LSTM was successful after CNN-LSTM with 0.943 R^2. CNN-LSTM was the most successful model with 0.961 R^2. Table 6 presents the experimental results for predicting hydro dam production.

Table 6. The experimental results for predicting lignite production

Model	RMSE	MAE	R^2
LR	178.74	88.14	0.899
RF	172.61	86.20	0.905
SVR	172.47	85.21	0.906
CNN	164.54	80.11	0.915
LSTM	143.25	69.84	0.934
CNN-LSTM	117.21	48.44	0.957

As seen in Table 6, all models except LR had R^2 above 0.9. RF and SVR gave close results. CNN-LSTM was the most successful model with R^2 of 0.957. Table 7 presents the experimental results for predicting wind power production.

Table 7. The experimental results for predicting wind power production

Model	RMSE	MAE	R^2
LR	181.12	135.77	0.990
RF	175.54	131.45	0.992
SVR	171.36	127.63	0.993
CNN	169.22	125.78	0.993
LSTM	157.56	115.87	0.995
CNN-LSTM	127.23	96.42	0.999

As seen in Table 7, all models had R^2 above 0.9. RF, SVR and CNN had similar results. LSTM had R^2 of 0.995 and CNN-LSTM had 0.999 R^2.

Figure 6 shows the comparative experimental results with respect to RMSE and MAE for energy production and consumption.

Figure 7 shows the comparative experimental results for energy production and consumption with respect to R^2.

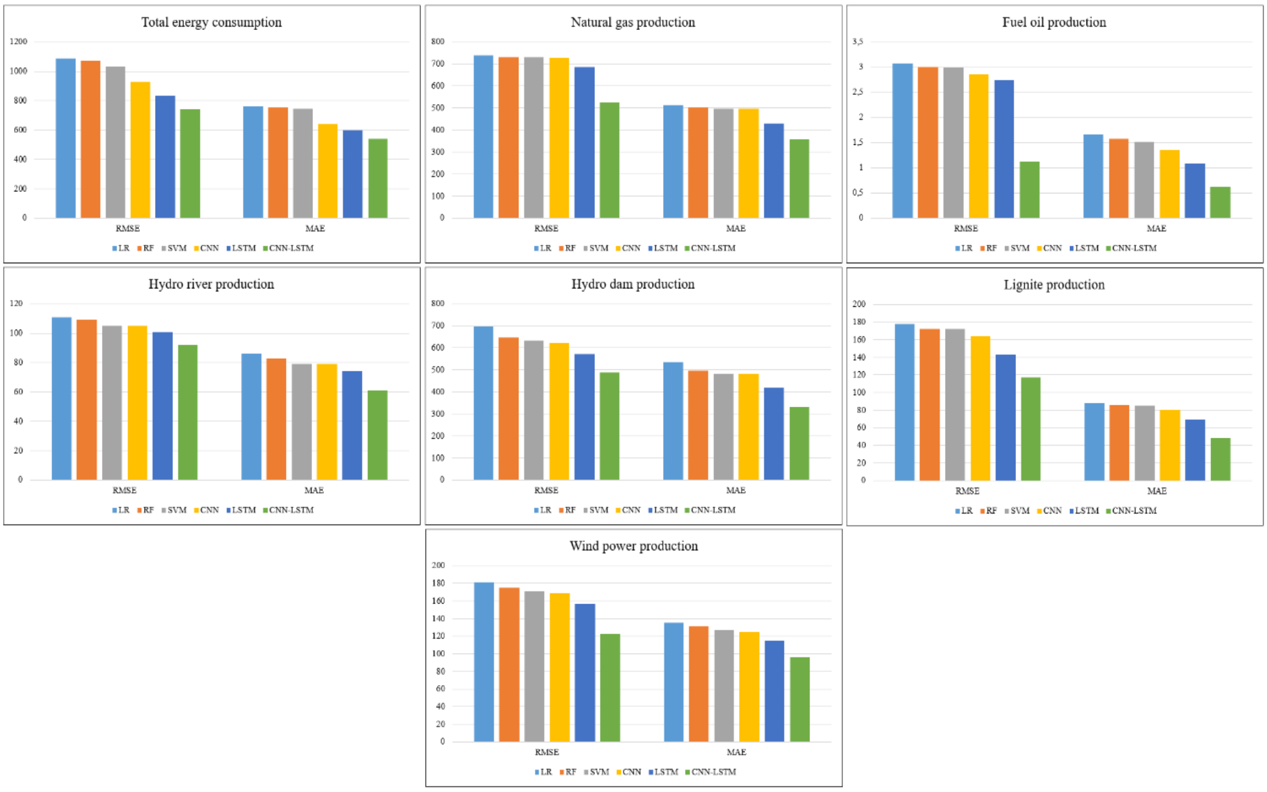


Figure 6. The comparative experimental results with respect to RMSE and MAE for energy production and consumption

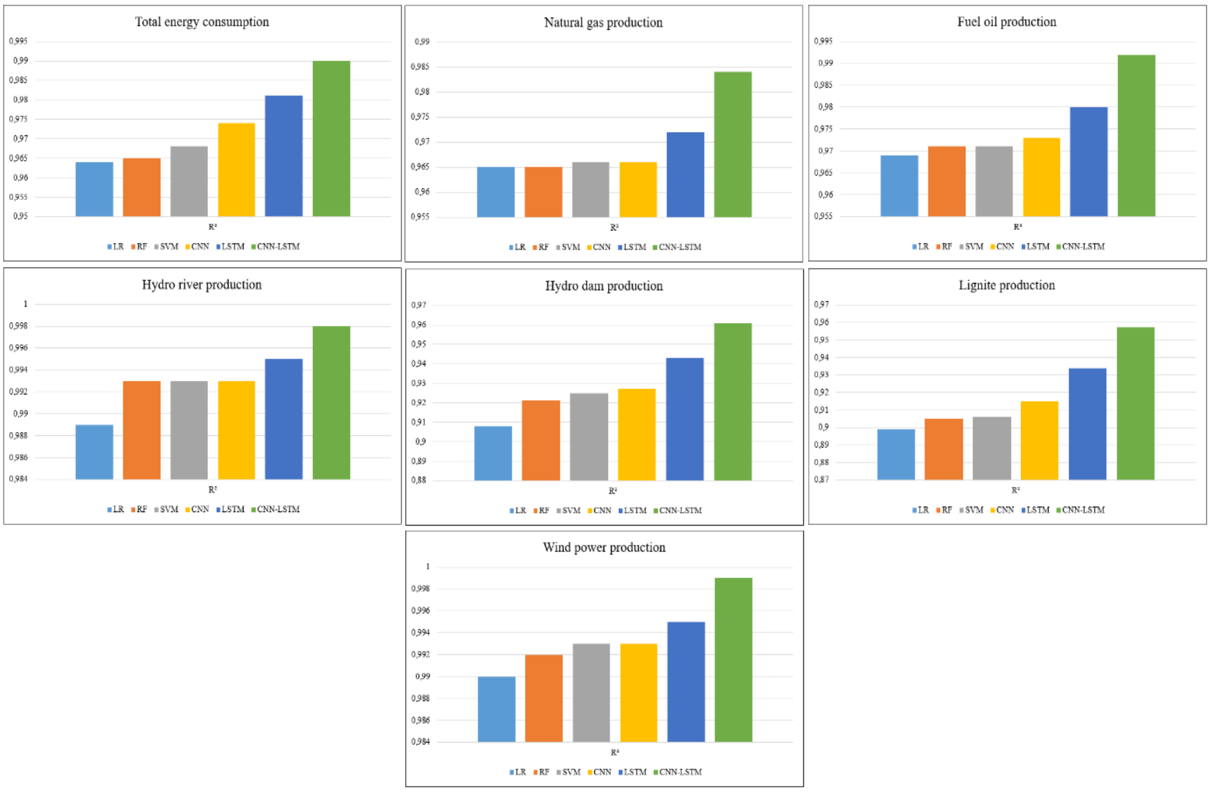


Figure 7. The comparative experimental results for energy production and consumption with respect to R^2

Experiments showed that CNN-LSTM outperformed the compared models for total energy consumption and production of natural gas, hydro dam, lignite, hydro river, wind and fuel oil, as seen in Figure 6 and Figure 7. CNN-LSTM was more successful than the compared models because CNN-LSTM enables the determination of spatial and temporal relationships thanks to its hybrid structure. CNN extracted local patterns in energy production and consumption data, while LSTM enabled the learning of short-term trends and long-term dependencies.

The success of LSTM can be explained by its capacity to capture long-term dependencies in time series data. Since the dataset used is dependent on external factors such as seasonal changes and energy demand, LSTM was able to successfully model the dependencies in the dataset. CNN is an effective model in capturing short-term patterns, but CNN's ability to capture long-term dependencies is limited. CNN was inefficient compared to CNN-LSTM and LSTM because it could not capture long-term relationships. SVR, RF and LR are not efficient enough in datasets such as complex and nonlinear time series. Especially in time series data, since the observation data are dependent on each other, traditional machine learning methods cannot capture long-term and short-term dependencies.

5. Conclusion

The objective of this study was to evaluate the efficacy of diverse machine learning and deep learning methodologies for energy forecasting, utilising hourly energy consumption and production data from across Turkey. Comprehensive experiments were conducted on LR, RF, SVR, CNN, LSTM, and the proposed hybrid CNN-LSTM model. Time series data were transformed into a regression problem using the sliding window method, allowing the models to learn temporal dependencies.

The findings of the experimental study indicated that the hybrid CNN with LSTM model exhibited superior performance in predicting total energy consumption and the production quantities of diverse energy sources in comparison to other models.

In particular, the CNN-LSTM model achieved the lowest error rates according to RMSE, MAE and the highest coefficient of determination according to R^2. This success is attributed to the combination of CNN's ability to identify local patterns and LSTM's capability to capture long-term dependencies in the data.

The findings highlight the effectiveness of deep learning-based hybrid models in energy forecasting. They significantly contribute to more accurate predictions of energy production and consumption, enabling more efficient use of energy resources and the development of improved energy management strategies.

Future studies can explore the model's generalizability to different energy markets and regions. Incorporating additional factors such as seasonal changes, economic indicators, and weather conditions into the model could enhance prediction accuracy. Moreover, applications like real-time forecasting and integration with smart energy management systems can be considered for practical implementations

References

[1] K. Amasyali and N. El-Gohary, "Machine learning for occupant-behavior-sensitive cooling energy consumption prediction in office buildings," *Renewable and Sustainable Energy Reviews*, vol. 142, p. 110714, 2021.

[2] S. Al-Dahidi, M. Alrbai, H. Alahmer, B. Rinchi, and A. Alahmer, "Enhancing solar photovoltaic energy production prediction using diverse machine learning models tuned with the chimp optimization algorithm," *Scientific Reports*, vol. 14, no. 1, p. 18583, 2024.

[3] J. Q. Wang, Y. Du, and J. Wang, "LSTM based long-term energy consumption prediction with periodicity," *energy*, vol. 197, p. 117197, 2020.

[4] Y. Wang, Z. Yang, L. Ye, L. Wang, Y. Zhou, and Y. Luo, "A novel self-adaptive fractional grey Euler model with dynamic accumulation order and its application in energy production prediction of China," *Energy*, vol. 265, p. 126384, 2023.

[5] B. Çelik and M. E. Çelik, "Root dilaceration using deep learning: a diagnostic approach," *Applied Sciences*, vol. 13, no. 14, p. 8260, 2023.

[6] Z. Dong, J. Liu, B. Liu, K. Li, and X. Li, "Hourly energy consumption prediction of an office building based on ensemble learning and energy consumption pattern classification," *Energy and Buildings*, vol. 241, p. 110929, 2021.

[7] S. K. Hora, R. Poongodan, R. P. De Prado, M. Wozniak, and P. B. Divakarachari, "Long short-term memory network-based metaheuristic for effective electric energy consumption prediction," *Applied Sciences*, vol. 11, no. 23, p. 11263, 2021.

[8] P. Nie, M. Roccotelli, M. P. Fanti, Z. Ming, and Z. Li, "Prediction of home energy consumption based on gradient boosting regression tree," *Energy Reports*, vol. 7, pp. 1246-1255, 2021.

[9] E. M. Al-Ali *et al.*, "Solar energy production forecasting based on a hybrid CNN-LSTM-transformer model," *Mathematics*, vol. 11, no. 3, p. 676, 2023.

[10] S. M. Malakouti, F. Karimi, H. Abdollahi, M. B. Menhaj, A. A. Suratgar, and M. H. Moradi, "Advanced techniques for wind energy production forecasting: Leveraging multi-layer Perceptron+ Bayesian optimization, ensemble learning, and CNN-LSTM models," *Case Studies in Chemical and Environmental Engineering*, vol. 10, p. 100881, 2024.

[11] M. Alaraj, A. Kumar, I. Alsaidan, M. Rizwan, and M. Jamil, "Energy production forecasting from solar photovoltaic plants based on meteorological parameters for qassim region, Saudi Arabia," *IEEE Access*, vol. 9, pp. 83241-83251, 2021.

[12] H. Yılmaz and M. Şahin, "Solar panel energy production forecasting by machine learning methods and contribution of lifespan to sustainability," *International Journal of Environmental Science and Technology*, vol. 20, no. 10, pp. 10999-11018, 2023.

[13] Y. Ledmaoui *et al.*, "Enhancing Solar Power Efficiency: Smart Metering and ANN-Based Production Forecasting," *Computers*, vol. 13, no. 9, p. 235, 2024.

[14] "Hourly energy data, Türkiye 2018-2023." https://www.kaggle.com/datasets/ahmetzamanis/energy-consumption-and-pricing-trkiye-2018-2023 (accessed 25/09/2024, 2024).

[15] X. Su, X. Yan, and C. L. Tsai, "Linear regression," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 4, no. 3, pp. 275-294, 2012.

[16] Y. Sun, F. Haghighat, and B. C. Fung, "A review of the-state-of-the-art in data-driven approaches for building energy prediction," *Energy and Buildings*, vol. 221, p. 110022, 2020.

[17] A. Utku and S. K. Kaya, "Deep learning based a comprehensive analysis for waste prediction," *Operational Research in Engineering Sciences: Theory and Applications*, vol. 5, no. 2, pp. 176-189, 2022.

[18] S. Jurado, À. Nebot, F. Mugica, and N. Avellana, "Hybrid methodologies for electricity load forecasting: Entropy-based feature selection with machine learning and soft computing techniques," *Energy*, vol. 86, pp. 276-291, 2015.

[19] P. Canbay and H. Tas, "Prediction of Heating and Cooling Loads of Buildings by Artificial Intelligence," *International Journal of Pure and Applied Sciences*, vol. 8, no. 2, pp. 478-489, 2022.

[20] R. Çekik and M. Kaya, "A New Performance Metric to Evaluate Filter Feature Selection Methods in Text Classification," *Journal of Universal Computer Science*, vol. 30, no. 7, p. 978, 2024.

[21] F. Zhang and L. J. O'Donnell, "Support vector regression," in *Machine learning*: Elsevier, 2020, pp. 123-140.

[22] Y. Liu, H. Chen, L. Zhang, X. Wu, and X.-j. Wang, "Energy consumption prediction and diagnosis of public buildings based on support vector machine learning: A case study in China," *Journal of Cleaner Production*, vol. 272, p. 122542, 2020.

[23] Y. Canbay, S. Adsiz, and P. Canbay, "Privacy-Preserving Transfer Learning Framework for Kidney Disease Detection," *Applied Sciences*, vol. 14, no. 19, p. 8629, 2024.

[24] A. Utku, E. D. Utku, and B. Kutlu, "Deep learning based an effective hybrid model for water quality assessment," *Water Environment Research*, vol. 95, no. 10, p. e10929, 2023.

[25] C. Lu, S. Li, and Z. Lu, "Building energy prediction using artificial neural networks: A literature survey," *Energy and Buildings*, vol. 262, p. 111718, 2022.
[26] F. Kuncan, Y. Kaya, Z. Yiner, and M. Kaya, "A new approach for physical human activity recognition from sensor signals based on motif patterns and long-short term memory," *Biomedical Signal Processing and Control*, vol. 78, p. 103963, 2022.
[27] Y. Kaya, Z. Yiner, M. Kaya, and F. Kuncan, "A new approach to COVID-19 detection from X-ray images using angle transformation with GoogleNet and LSTM," *Measurement Science and Technology*, vol. 33, no. 12, p. 124011, 2022.
[28] Y. Kaya, M. Kuncan, E. Akcan, and K. Kaplan, "An efficient approach based on a novel 1D-LBP for the detection of bearing failures with a hybrid deep learning method," *Applied Soft Computing*, vol. 155, p. 111438, 2024.
[29] G. Li, F. Li, C. Xu, and X. Fang, "A spatial-temporal layer-wise relevance propagation method for improving interpretability and prediction accuracy of LSTM building energy prediction," *Energy and Buildings*, vol. 271, p. 112317, 2022.

Journal of Soft Computing and Artificial Intelligence

Journal homepage: https://dergipark.org.tr/en/pub/jscai

International Open Access

Volume 05
Issue 02

December, 2024

Research Article

Review Mate: A Cutting-Edge Model for Analyzing the Sentiment of Online Customer Product Reviews using ML.NET

Özge CÖMERT[1], Nurcal YÜCEL[2]

[1] *Department of Transportation Services, Samsun University, 55000, Samsun, Turkey*
[2] *Department of Business Administration, Fırat University, 23000, Elazığ, Turkey*

ARTICLE INFO

Article history:
Received December 3, 2024
Revised December 12, 2024
Accepted December 17, 2024

Keywords:
Sentiment analysis,
Machine learning,
Classification,
ML.NET

ABSTRACT

E-commerce has become increasingly important in recent years due to several factors such as convenience, global reach, lower costs, personalization and uninterrupted access. In e-commerce, product reviews by customers can significantly impact purchasing behavior by providing social proof, establishing trust, aiding decision-making, improving search engine optimization, and increasing sales. Conducting an evaluation of the primary impacts of customer reviews on purchasing behavior through automated machine learning techniques has the potential to facilitate the advancement of diverse online business models. In this scope, we come with a new machine-learning model for evaluating customer sentiment based on product reviews. To this aim, a dataset consisting of 1000 positive and 1000 negative customer reviews was created by collecting publicly shared comments from online shopping websites serving in Turkey with a data collection tool developed by our research group. The model development was carried out on ML.NET, an open-source and cross-platform machine learning framework. In order to reach the most efficient model, a total of 36 machine learning models were explored for the solution of the problem within the scope of the experimental study. As a result, the model named Lbfgs Logistic Regression Binary was found to be the most efficient. The related model provided an accuracy rate of 94.76%. An API service called Review Mate has been developed to expand the potential impact of the proposed machine learning model and enable its use in different online business models. According to the findings, the proposed method outperforms the previous approach in terms of classification performance and also provides avenues for the discovery of new product ideas.

1. Introduction

Over the past few decades, the internet has revolutionized the way we live our lives and conduct business. E-commerce, in particular, has played a crucial role in this transformation. With the ability to shop from anywhere, at any time, e-commerce has expanded our reach to customers all around the world [1]. This increased access to a global marketplace has opened up countless opportunities for businesses to grow and succeed [2]. In addition, e-commerce has made it easier for consumers to find and purchase products that suit their needs and preferences. As technology continues to advance, the future of e-commerce looks bright, with even more potential for growth and innovation [3].

In order to stay competitive and meet the evolving needs of their customers, companies are increasingly focused on gathering and analyzing feedback. By

* Corresponding author
e-mail: ozge.comert@samsun.edu.tr
DOI: 10.55195/jscai.1595798

collecting customer comments and reviews, businesses gain valuable insights into the strengths and weaknesses of their products and services [4].

Armed with this information, companies can make strategic improvements, from minor tweaks to major overhauls, to better meet the needs of their target audience. Whether it's improving the functionality of a website, streamlining a checkout process, or introducing a new product line, companies that actively listen to their customers and incorporate their feedback into their operations are better positioned for success [5]. By prioritizing customer satisfaction, businesses can build brand loyalty, improve their reputation, and drive growth in the competitive marketplace.

Recently, it is seen that artificial intelligence-supported studies have been carried out to draw meaningful conclusions from customer comments [6]. These studies focus more on customer comments instead of using customer transaction databases containing demographic, psychographic, and purchasing behavior information traditionally used [7]. A framework called GSITK has been proposed for performing a wide range of sentiment analysis tasks such as dataset acquisition, text preprocessing, model design, and performance evaluation. It is intended for both researchers and practitioners and offers implementations of common tasks as well as facilitates the replication of previous sentiment models [8].

This study proposes an interpretable machine learning approach to customer segmentation for new product development, using online product reviews. Interpretable machine learning identifies non-linear relationships with high performance and transparency. Customer segmentation using the proposed approach outperforms a previous approach based on emotions. The findings demonstrate superior clustering performance and opportunities for new product concepts [9]. Research has been carried out to collect consumer feedback for airline service reviews and perform various forms of analysis on them. The study proposes a classification approach using machine learning techniques to identify positive and negative words or comments on a text-oriented database. Several ML algorithms such as Naive Bayes, Support Vector Machine (SVM), Decision Tree (DT), and Google BERT model are used to test the performance of sentiment analysis. In term of performance criteria such as accuracy, precision, recall, and F1-score, it is discovered that BERT outperforms other ML techniques [10].

A sentiment analysis was conducted using a substantial corpus of textual data from individuals commenting on a range of services and products. This analysis led to the formation of new datasets in Turkish, English, and Arabic, marking the first occasion on which comparative sentiment analysis has been conducted on texts in these three languages. Furthermore, the researchers were presented with a comprehensive study comparing the performances of both pre-trained language models and deep learning and machine learning models for Turkish, Arabic, and English. [11]. Other research has looked at customer satisfaction with baby products on Amazon. This was achieved through text mining and survey-based methodologies. The research model was developed based on factors derived from text mining, and a questionnaire was distributed and analyzed using Partial Least Squares Structural Equation Modelling (PLS-SEM). The results revealed that a number of dimensions influence customers' experiences of baby products. [12]. A different study proposes a CNN and LSTM-based supervised deep learning classifier for the multi-class analysis of sentiment in Bengali social media posts. These posts, which have been labelled according to their content, may be sexual, religious, political or acceptable. Six machine learning models have been developed as the basis for this classifier, with two different feature extraction techniques. These have been evaluated on a labelled dataset of 42,036 Facebook comments, with an accuracy of 85.80% achieved by the CLSTM architecture [13]. In a separate study, a dataset called Abusive Turkish Comments (ATC) has been proposed for the detection of abusive comments in Turkish on Instagram. In this study, sentence-level sensitivity annotation is performed. The performance of five well-known classifiers (Naïve Bayes, SVM, DT, RF and LR) and two re-weighted classifiers (i.e., Adaptive Boosting (AdaBoost), eXtreme Gradient Boosting (XGBoost)) is evaluated in terms of F1-score, precision and recall. Ultimately, the findings indicated that the CNN model exhibited the most optimal performance on the oversampled ATC dataset [14]. Another study identifies sentiments in social media texts using machine learning. E-commerce website reviews and product reviews have been transformed into a tabular format for machine learning-based sentiment analysis. The reviews have been classified into three categories: positive, negative, and neutral. This classification was based on review scores. Turkish sentiment analysis models have been developed using SVM, RF, DT, LR, and k-nearest neighbors (KNN). The cross-validation results on independent test data from the same e-commerce website demonstrate that the SVM-based and RF-based sentiment analysis models outperform the others. [15]. A further study proposes the development of a new optimized machine learning (ML) algorithm, designated the Local Search Improved Bat Algorithm Based Elman Neural Network (LSIBA-ENN), for the analysis of sentiment in online product reviews. The Web Scrapping Tool (WST) is used to collect customer reviews from e-commerce websites. The web scraping data is preprocessed and then classified into positive, negative and neutral categories using LSIBA-ENN. This process demonstrated that LSIBA-ENN exhibited the most optimal performance [16]. In other related work, an innovative computational intelligence framework for the efficient prediction of customer review ratings is presented. Specifically, the proposed framework integrates Singular Value Decomposition (SVD), dimensionality reduction techniques, Fuzzy C-Means (FCM), and Adaptive Neuro-Fuzzy Inference System (ANFIS). The output of the proposed approach exhibited high accuracy. Additionally, the results indicated that in the case of large datasets, only a subset of the data is necessary for the construction of a system for predicting the evaluation scores of textual reviews [17].

In this study, we have tried to make our model robust by selecting the most effective methods and combining the best ideas from related works. We also believe that we have introduced innovations that have the potential to contribute to the field.

The main contributions of this study to the related field can be listed as follows:

(1) In this study, we created a large dataset using online customer product reviews on e-commerce sites.
(2) We made the dataset open-access and shared the dataset as a downloadable material in CSV format.
(3) We approached the light preprocessing steps differently from other studies on sentiment analysis in Turkish and took care to leave the comments as much as possible without affecting the raw data.
(4) Overall, we explored a total of 36 machine learning model within the scope of our study and selected the algorithm that produced the best results for distribution.
(5) We moved the proposed model to an API service and turned it into a software that can be used with zero configuration.
(6) We shared the service and AI model we developed in a GitHub repository as open access. We found that the model we proposed can be adapted to different business models and can be easily used to draw meaningful conclusions from product reviews.

The rest of the study is organized as follows: Section 2 summarizes the dataset and methodology. Section 3 presents the findings and provides a discussion. Finally, Section 4 presents the concluding remarks.

2. Material and Methods

2.1. Dataset

Within the scope of the study, collecting customer reviews is an essential step for model training. In order to accomplish this step, real customer reviews on the well-known websites hepsiburada.com, n11.com and trendyol.com in Turkey were taken into consideration.

A data collection tool called "*Product Pulse*" was developed to collect and process the customers' reviews. This tool was built on the ASP.NET Core MVC project template and uses the MSSQL database to provide persistent storage of data.

Table 1 Distribution of comments according to the source collected

	Positive	**Negative**	**Total**

hepsiburada.com	174	109	283
n11.com	253	705	958
trendyol.com	573	186	759
Total	1000	1000	2000

Table 1 shows the distribution of the data collected as a result of an extensive study. As can be seen, a total of 283 comments were received from hepsiburada.com, 958 comments from n11.com and 759 comments from trendyol.com respectively. A total of 2000 online customer product reviews, both positive and negative, were collected to be used in model training by paying attention to the balanced distribution of the data.

The data set produced within the scope of the research can be downloaded from the GitHub repository where experimental studies are shared.

2.2.Preprocessing

The proposed model is intended to work directly on customer reviews and contribute to the development of various online business models. For this reason, we make sure that customer reviews are as raw as possible. For this reason, in the preprocessing step, we remove spaces at the beginning and end of sentences. We eliminate phrases that use quotation marks. We eliminate start-of-line (\r) and end-of-line (\n) escape expressions and convert all characters to lower case before feature extraction.

Cleaning and standardizing text data before analysis is important for obtaining more accurate and reliable results. Removing spaces ensures that the text is more consistent and readable by eliminating unnecessary spaces at the beginning and end of sentences.

Removing quotation marks simplifies the text by recognizing that quotation marks in texts may be unnecessary for some analysis processes and could complicate the analysis process.

Removing newline escape expressions ensures that unwanted errors that may occur during text data processing are eliminated, resulting in the text being processed properly. Converting all characters to lowercase ensures that the analysis process is more consistent in cases where case sensitivity may be important in text analysis processes.

These preprocessing steps contribute to cleaning and standardizing text data before analysis, resulting in more accurate and reliable results [18].

2.3.Machine Learning

Typically, the process of building a model using machine learning algorithms involves a few basic steps such as problem statement, feature engineering, obtain dataset, feature extraction, model training and model evaluation [19].

Firstly, data needs to be collected and pre-processed to ensure that it is clean, complete and in the correct format [20]. Then, an appropriate machine learning algorithm must be selected based on the nature of the problem and the type of data being used. The algorithm must then be trained using a training dataset, which involves feeding it with inputs and expected outputs to enable the algorithm to learn patterns and relationships in the data [21].

After the algorithm has been trained, it can be tested using a separate test dataset to assess its accuracy and performance [22]. The model can be further refined and improved by changing parameters or using different algorithms, and it may be necessary to repeat the training and testing process several times to achieve the desired results.

After fine tuning and validating the model, it can be used to make predictions or generate insights based on new data inputs. The model may need to be continuously monitored and improved to ensure that it remains accurate and effective over time.

2.4.ML.NET

The Microsoft ML.NET framework was announced in May 2018. The framework is published on the GitHub platform as open access under the MIT license.

The Microsoft ML.NET library is already used in Chart Decision in Excel, Slide Design in Power Point, Windows Hello and Azure Machine Learning products.ML.NET is designed to be intuitive for experienced .NET developers. At the heart of ML.NET is the *MLContext* object, a singleton class like *AppContext*.

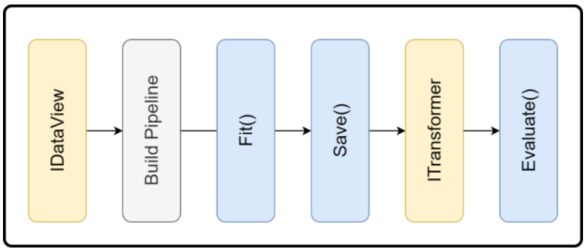


Figure 1 The high-level architecture of ML.NET

The *MLContext* object provides access to the entire catalog of trainers that ML.NET offers, such as anomaly detection, binary classification, clustering, regression and time series [23].

Fig. 1 shows the high-level architecture of ML.NET framework. Each block in the architecture can be summarized as follows: *IDataView* manages data loading, cleaning, and transfer in ML.NET, facilitating efficient and scalable operations. *BuildPipeline* creates sequential data processing and training chains, streamlining workflow construction. *Fit* trains datasets with specific algorithms to establish relationships between features and targets. Save stores trained models for easy retrieval and reuse. *ITransformer* enables result extraction and application, while *Evaluate* assesses model performance, guiding decision-making processes. Together, these components enhance ML.NET's usability, efficiency, and reliability in machine learning operations.

2.5. Proposed Model and Service

The general block diagram of the developed application is shown in Fig. 2. In the following sections, the details of the related blocks are explained in relation to the literature. Details of the first block related to data acquisition are already presented in Section 2.1.

As for *ReviewMateLib*, this is an ASP.NET Core library (*classlib*) project. Basically, the model data, training, evaluation, consumption and distribution of the generated model all take place within this library project. This project mainly consists of two main blocks. The first one performs the model training function, while the second one focuses on model consumption. In the model training phase, a pipeline for model creation, training and evaluation is built. The following steps are followed in the pipeline: In first step, *FeaturizeText* function is used to convert text data into numeric vectors. This function uses techniques such as bag-of-words model, n-gram features, TF-IDF criteria and similar techniques to make the text attributes suitable for model training. *This* function extracts a set of features for representing text data in vector space. This allows

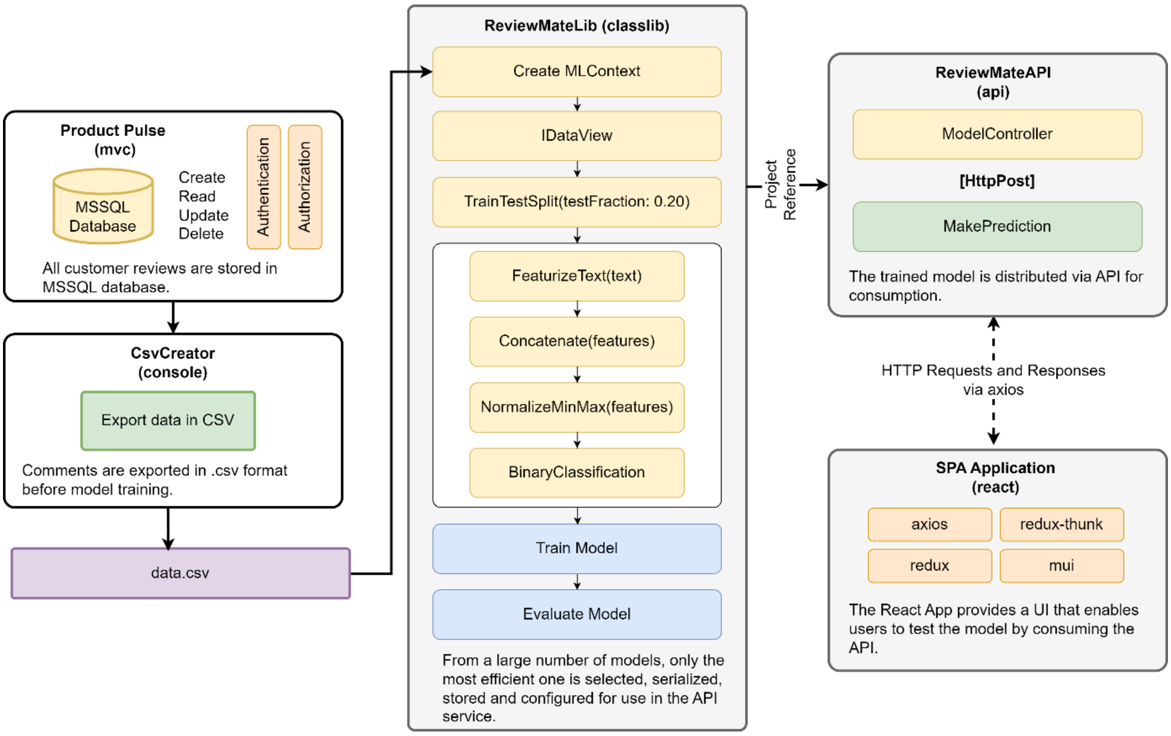


Figure 2 General block diagram of the study

text data to be represented by numeric values that can be used in machine learning algorithms. This function is a tool of ML.NET used in natural language processing (NLP) and is an important step to convert text data into numeric data. *Concatenate* in ML.NET performs a column concatenation operation and combines the specified columns into a single column.

Normalization is a process that brings the values in a dataset to a scale range. This process is used to balance the effects of variables with different scales, improve the performance of optimization algorithms, reduce the risk of overfitting, and increase the convergence speed of optimization. It ensures that the model learns more balanced and stable results. *NormalizeMinMax* takes a specific column of features and rescales all values to a specific range (usually [0,1] or [-1,1]). This function can help outputs perform better in a model. It is especially useful in cases where an un-normalized feature column has a larger range of values in terms of scale. This function can be used to pre-process data and can be combined with other transformations as part of an ML.NET pipeline. In Eq. (1), y represents the normalized value, which is between 0 and 1. x is the normalized feature.

$$y = \frac{(x - min)}{(max - min)} \quad (1)$$

In the classification step, binary classification models with different hyper-parameters are considered to obtain the most effective model.

Basically, four different machine learning algorithms are used for this purpose. These models are LbfgsLogisticRegression, FastTree, SdcaLogisticRegression, and LightGbm respectively.

Based on varying hyperparameter values, a total of 36 machine learning models are explored in this study. For a more concise and understandable presentation, only the Logistic Regression model that provides the best generalization performance is presented in the next section.

2.6. Logistic Regression

Logistic regression (LR) is a classification algorithm used particularly when the dependent variable (outcome) is categorical. It is commonly used in binary classification problems.

This method predicts the probability of the dependent variable belonging to a certain category using a linear combination of independent variables (predictors). Essentially, logistic regression classifies data appropriately using a sigmoid function called the logistic function. By fitting the model to the data, it predicts the probability of belonging to a specific class.

Linear regression is a type of linear model and it matches feature vector $\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}$ to a scalar as described in Eq. (2).

$$p(Y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}} \quad (2)$$

Herein, $p(Y = 1 \mid X)$ represents the probability of Y being 1. e is the base of natural logarithm. $\beta_0, \beta_1, \ldots \beta_n$ are the coefficients of the model and $X_1, X_2, \ldots X_n$ are the independent variables.

The model training is completed in accordance with the mentioned steps. From this point on, the trained model can be utilized. In this context, the *CreatePredictEngine* function loads a trained model and creates a prediction engine to make predictions using this model.

The prediction engine takes input data and calculates outputs based on the model created during training.

The predictive engine uses only the memory required for the prediction function and therefore works efficiently on large datasets. In the final step, the *Predict* function executes the prediction function based on the inputs using *PredictEngine*.

2.7. Model Evaluating

Several metrics are used to evaluate the performance of classification models. These metrics include accuracy, precision, recall, F1 score, ROC curve and AUC.

These metrics are based on the classifications of true positive (TP), false positive (FP), true negative (TN) and false negative (FN). In our case, TP and TN represent the number of positive and negative reviews that were actually correctly detected, respectively.

Table 2 Performance metrics for evaluation classification models

Measure	**Formula**	**Short Description**
Accuracy (Acc)	$\frac{TP+TN}{TP+FP+FN+TN}$	In machine learning, accuracy is a performance metric used to measure the number of correctly classified instances out of the total instances in a dataset. It provides a quick and easy way to evaluate the effectiveness of a classification model.
Sensitivity (Se)	$\frac{TP}{TP+FN}$	Precision, also known as recall or true positive rate, measures the proportion of true positive cases correctly identified by a classification model. Precision is particularly important in applications where the cost of a false negative (missing a positive case) is high.
Specificity (Sp)	$\frac{TN}{TN+FP}$	The specificity is a performance measure used to assess the ability of a classification model to correctly identify negative examples. It is a complement to sensitivity, which measures the model's ability to correctly identify positive examples.
F1-Score	$\frac{2TN}{2TP+FP+FN}$	The F-score is a performance measure that combines precision and recall into a single value and provides a balanced measure of a model's accuracy. It is the harmonic mean of precision and recall, with 1 indicating excellent precision and recall and 0 indicating poor performance.

Similarly, FP and FN match the number of positive and negative reviews detected as false. The correct interpretation and use of these metrics ensures that the performance of the model is not misleading and provides guidance for model improvement [24]. Table 2 summarizes the several performance metrics derived from the confusion matrix.

2.8. Model Distributing

The ASP.NET Core API project template is used to build RESTful web services. This template enables developers to provide a presentation layer where they can create, read, update and delete data using the HTTP protocol [25]. It is typically used for sending and receiving data in JSON format. The template provides developers with various features in the API such as routing, input validation and CORS settings, and can be easily customized for additional features such as JWT authentication. ReviewMateAPI is used to allow the trained model to be easily consumed by different applications. The MakePrediction post function defined in the ModelController allows different applications to consume the related service [26].

2.9. Single Page Application

Single Page Application (SPA) applications are web applications where all pages are rendered in the browser and not by the server. This allows for a faster and more dynamic user experience. React is one of the popular JavaScript libraries used in the development of SPA applications [23]. React has a component-based structure and automatically manages changing data and updates the user interface. React also offers many great tools and features such as Virtual DOM, JSX, React Router, Context API, Redux. These features provide developers with a powerful toolset for building and managing SPAs. A React application including axios, redux, redux-thunk and Material UI (mui) libraries was developed from the scope of work to test our model live on a user interface.

Figure 3 Visualizing the essence of the dataset through a word cloud

3. Results and Discussion

As a result of the study, a dataset consisting of a total of 2000 online customer product reviews was created. In order to support similar studies and to ensure that the proposed algorithms can be easily compared, the dataset has been shared as open access in .csv format. In Fig. 3, the essence of the dataset is visualized through a word cloud. A word cloud graph is used as a visual representation of words in text data. The visibility of words indicates how frequently they are used in the text.

All experimental studies were carried out on a local workstation with Windows 11 64-bit operation system and Intel© Xeon© Gold 6132 CPU 2.60 GHz. The memory was 131 GB. Depending on the variation of the hyperparameter values, a total of 36 different models were explored to solve the problem. The most efficient models with values of hyperparameters are presented in Table 3.

Table 3 is important for presenting the results of the experimental study conducted. Designed to compare the performance of 36 different models using various hyperparameter values, this table provides a detailed description of the hardware and software environment used in the study. With this information, the table allows for the identification of the most effective models and clearly determines the hyperparameter values associated with these models. These details are important for interpreting the results of the study and guiding future research endeavors.

The hold-out validation technique was used instead of the k-fold cross-validation technique as there was sufficient data for model training. In this context, 80% of the data was reserved for the training set, while the remaining 20% was used as the test data set.

Table 3 Hyperparameters with values for the most efficient models

Models	Parameters	Values
LbfgsLogistic Regression	L1 Regularization	0.03125
	L2 Regularization	15.4891
	Tolerance	1E-8
	Enforce Non-Negativity	false
FastTree	Number of Leaves	20
	Number of Trees	100
	Minimum Example Count per Leaf	10
	Learning Rate	2
	Shrinkage	1
	Categorical Split	Categorical Split.Auto
	Feature Fraction	1
SdcaLogisticRegression	L1 Regularization	0.0
	L2 Regularization	1.0
	Maximum Number of Iterations	100
	Convergence Tolerance	0.0001
	Shuffle	True
	Optimization Tolerance	1E-8
LightGbm	Learning Rate	0.1
	Number of Leaves	20
	Number of Iterations	100
	Minimum Example Count per Leaf	20
	Maximum bin Count per Feature	256

In line with the aforementioned models and parameters, 80% of the dataset was used as the training set. The training stage was completed in 237.19 seconds and a total of 36 model discoveries were completed. Actually, these 36 models are basically based on four regression models given in The confusion matrices of the models regarding the classification results are given in Fig. 4.

The LbfgsLogisticRegression algorithm produced 94.76% of accuracy, 92.39% of sensitivity, 97.06% of specificity, and 94.55% of F1-score, respectively.

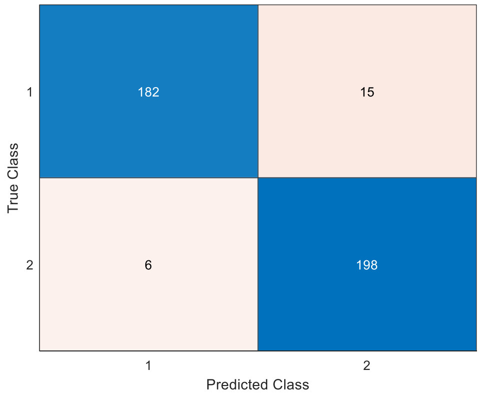
This model provided the best generalization performance. Similarly, the SdcaLogisticRegression algorithm provided satisfactory results. This algorithm yielded 93.77% of accuracy, 93.91% of sensitivity, 93.63% of specificity, and 93.67% F1-score. The results for other algorithms are given in Table 4.

business models. In this context, the results were considered promising and satisfactory.

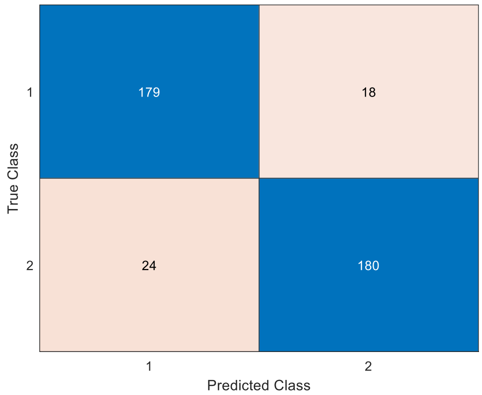
When the related studies in the literature are examined, it is seen that almost all of the studies follow the steps of data collection, preprocessing, model development and model evaluation and the

Table 3 The classification results of the most efficient models with performance metrics

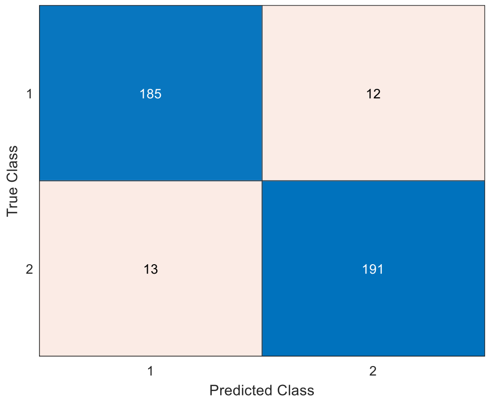
Trainer	Acc	Se	Sp	F1-Score	AUC
LbfgsLogisticRegression	0.9476	0.9239	0.9706	0.9455	0.9883
FastTree	0.8953	0.9086	0.8824	0.8950	0.9656
SdcaLogisticRegression	0.9377	0.9391	0.9363	0.9367	0.9833
LightGbm	0.9152	0.9289	0.9020	0.9150	0.9694



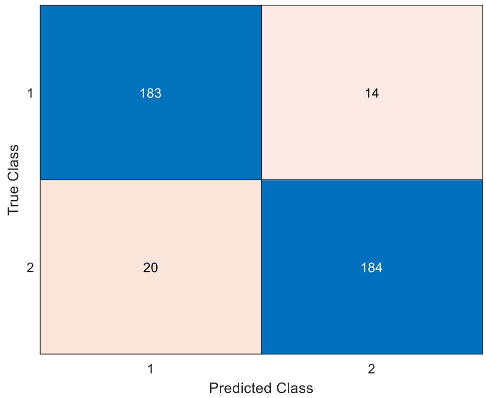
(a) LbfgsLogisticRegression



(b) FastTree



(c) SdcaLogisticRegression



(d) LightGbm

Figure 2 Confusion matrices of the models

The model development and evaluation studies revealed that the proposed model can automatically classify Turkish online customer product reviews with 94.76% of accuracy, 94.55% of F1-score and 0.9883 of AUC scores. The results show that the proposed model can be used in the innovative online

studies are concluded in this manner. In this study, we developed an API service to extend the domain of our proposed model, to support online business development processes and to test the model with zero configuration. Moreover, we developed a SPA application with React JavaScript UI Framework to enable quick testing of the developed model and the model can be downloaded in the GitHub repository. The link of mentioned application is shared at end of the work.

In the other case, the Thunder Client VS Code Extension was used as an API testing tool. An HTTP Post Request was sent directly to the API service and the generated response was received. The service

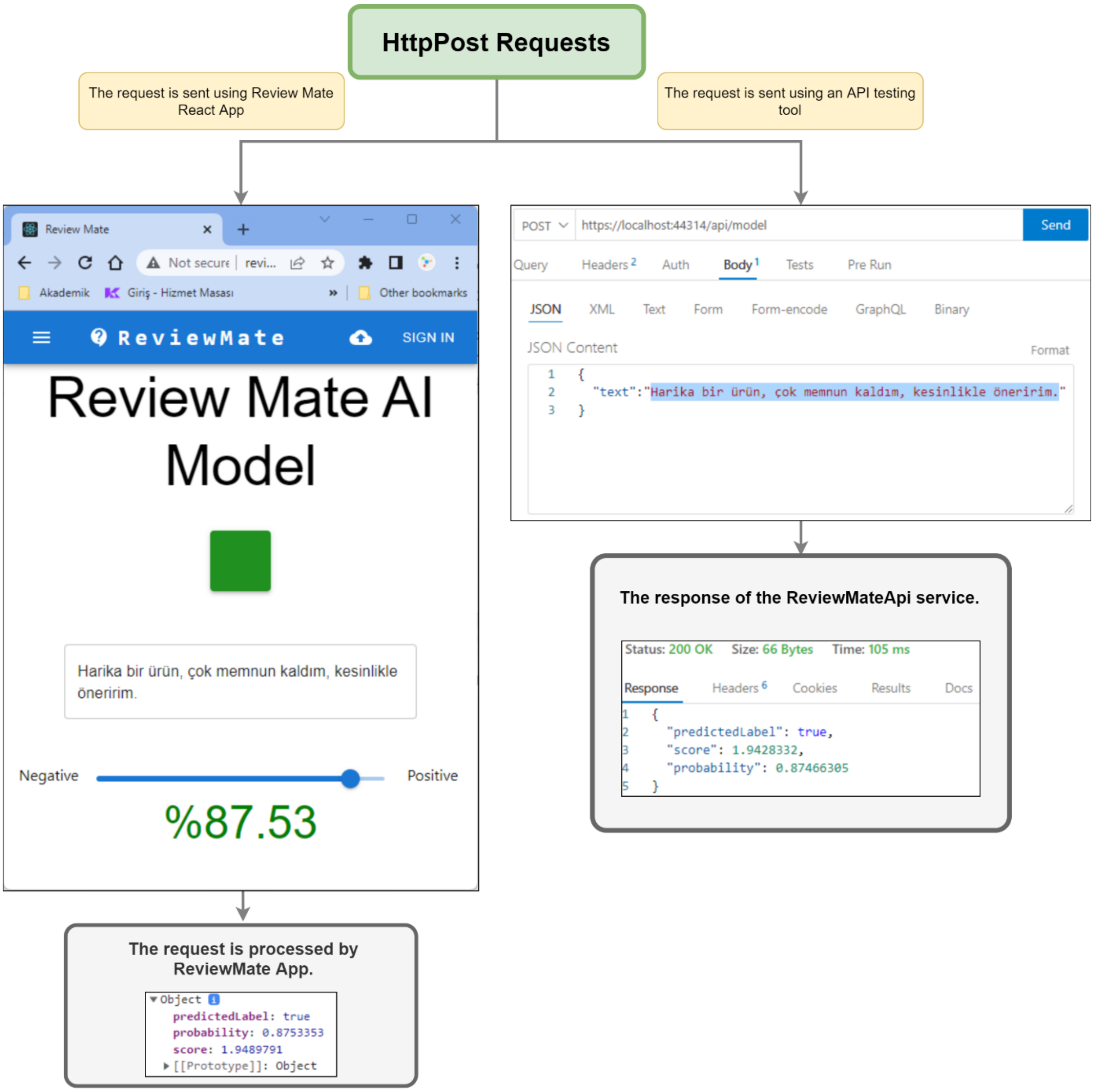


Figure 3 Analyzing sentiment of online customer product reviews with Review Mate Application

The overall diagram of the system designed in previous Fig. 2 is provided. The screenshot of the SPA application in this diagram is given in Fig. 5. Fig. 5 visualizes the processing of Http Post requests through the ReviewMate React App and ReviewMate API service, respectively. As seen in the figure, the ReviewMate React App performs sentiment analysis by reacting directly to customer reviews. At this point, the ReviewMate React App consumes API service and generates the responses simultaneously while the end user enters a comment.

response generated for requests includes three fields: predicted label, probability and score. The same comment was sent to the service as two separate requests and in both cases the requests were classified as positive. However, although the results are quite close, the responses are not equal. This can be considered normal in the context of how machine learning algorithms work.

In Table 5, the generated responses of the proposed model for customer product reviews randomly

Table 4 Randomly selected customer reviews and the responses of the ReviewMate API

Reviews	**Language**	**Translated to Turkish using Google Translate**	**Scale**	**Predicted Class**	**Probability**
Thanks for an excellent product.	English	mükemmel bir ürün için teşekkürler.	5*	+	0.9531
Great iPad, never had a problem with it but I wish I would have order it with the wifi and internet feature but it's a good product and it was amazing for gaming and school	English	Harika iPad, hiç sorun yaşamadım ama keşke onu wifi ve internet özelliğiyle sipariş etseydim ama iyi bir ürün ve oyun ve okul için harika	5*	+	0.7519
Entregou tudo certo, bem antes do prazo combinado.	Portuguese	Kararlaştırılan son tarihten çok önce her şeyi doğru bir şekilde teslim etti.	5*	+	0.8258
O produto veio com defeito, a bateria só carrega até 42%.	Portuguese	Ürün kusurlu geldi, pil sadece %42'ye kadar şarj oluyor.	2*	-	0.2227
The touchscreen went bad after a few days. Had to keep tapping and dragging before it worked. It continually asks if I want to upgrade, you have to log in all the time, even though I just did that. I couldn't just return it, I had to call in to explain that I did the obvious to try to fix it. Return experience horrible.	English	Dokunmatik ekran birkaç gün sonra bozuldu. Çalışmadan önce dokunup sürüklemeye devam etmem gerekiyordu. Sürekli olarak yükseltmek isteyip istemediğimi soruyor, ben bunu yapmış olmama rağmen her zaman oturum açmalısın. Öylece iade edemedim, düzeltmek için bariz olanı yaptığımı açıklamak için aramak zorunda kaldım. Geri dönüş deneyimi korkunç.	1*	-	0.1437
Great I pad with a lot of power! I love it	English	Çok fazla güçle harika bir yastıklama yapıyorum! Bayıldım	5*	+	0.8648
Calidad precio 10/10	Spanish	Kalite fiyatı 10/10	5*	+	0.5352
ste Ipad es una de las mejores compras que he podido haber hecho, escogí el modelo de 64 GB y es suficiente si lo que buscas es ver series, películas, navegar por la red o incluso jugar, corre perfectamente Genshin Impact, LOL Wild Rift, o Brawls Stars, la potencia es suficiente para un usuario que busca un medio de entretenimiento, a parte la batería dura todo el día, sumamente satisfecho con este Ipad.	Spanish	Bu Ipad, yapabileceğim en iyi satın alımlardan biri, 64 GB modelini seçtim ve aradığınız şey dizi, film izlemek, internette gezinmek ve hatta oyun oynamaksa yeterli, mükemmel çalışıyor Genshin Impact, LOL Wild Rift veya Brawls Stars, gücü eğlence aracı arayan bir kullanıcı için yeterli, ayrıca pili tüm gün gidiyor, bu iPad'den son derece memnun.	5*	+	0.7308
特殊な素材を使ってるのか？ゴム部分が臭く感じました。耐久性使いやすいさは 100 点かな！	Japanese	Özel malzemeler mi kullanıyorsunuz? Kauçuk kısmın kokusunu alabiliyordum. Dayanıklılık ve kullanım kolaylığı 100 puan!	4*	+	0.1328
背面の立てかける部分が最初から開きにくく、数回開閉したら根本から折れてしまいました。プラスチックではなく金具か何かにした方が良いと思います	Japanese	Arkadaki parçayı baştan açmak zor oldu bir kaç kez açıp kapadıktan sonra tabandan kırıldı. Bence plastik yerine metal bağlantı parçaları veya başka bir şey kullanmak daha iyi	1*	-	0.1955

selected from Amazon.com source in different languages are examined. In this context, customer product reviews of variable length in English, Portuguese, Spanish and Japanese languages are considered.

While evaluating the reviews, the stars given by the customers were taken into account. The foreign language reviews were translated into Turkish with Google Translate and this translation was directly applied as input to the model. The class and probability value predicted by the proposed model are presented. As a result, the evaluation results of customer product reviews translated into Turkish from different languages were found to be satisfactory.

In Table 5, the generated responses of the proposed model for customer product reviews randomly selected from Amazon.com source in different languages are examined. In this context, customer product reviews of variable length in English, Portuguese, Spanish and Japanese languages are considered. While evaluating the reviews, the stars given by the customers were taken into account. The foreign language reviews were translated into Turkish with Google Translate and this translation was directly applied as input to the model. The class and probability value predicted by the proposed

Table 5 Comparison of related studies

Methods	**Description dataset**	**# of samples**	**# of classes**	**Availability**
Hybrid deep learning model, feature selection [27].	Prompt Cloud dataset	400.000 samples	5	Private dataset.
Multi attention and Bidirectional GSNP model (MA-BiGSNP) [30].	SST-2, IMDB, MR, Twitter, AirRecord	Over 6.000 samples	V	Public datasets
Machine learning techniques [28].	Hotes Amazon	Four different datasets with various numbers	V	Private datasets
Cluster based classification model [29].	iPod	1.811	2	Private dataset
A heterogeneous network model [31].	IMDB Yelp 2013 Yelp 2014	Over 78.000	V	Public datasets
Machine learning and deep learning models [11].	Movie, Game , Small appliances, Technological products, Large home appliances	100.000	3	Private dataset
Pretrained word embeddings and deep learnings [32].	Two different datasets produced	1.5 million samples	10	Private datasets
Machine Learning techniques [15].	hepsiburada	31.750 samples	3	Private dataset
Embedding based deep learning [18].	hepsiburada, n11, trendyol	2000 samples	2	Public dataset
This work, Lbfgs Logistic Regression model and Machine Learnig .NET.	hepsiburada, n11, trendyol	2000 samples	2	Public dataset

model are presented. As a result, the evaluation results of customer product reviews translated into Turkish from different languages were found to be satisfactory.

Table 6 provides a comprehensive comparison, taking into account similar studies in literature. It should be noted that conducting a direct one-to-one comparison may not be feasible due to variations in methods, samples, number of classes, and distribution differences in the datasets, as well as the utilization of different software and hardware resources. In this context, preferred environments for experimental studies, datasets used, sample sizes, availability of studies for open access, methods, and performance metrics have been considered.

The majority of datasets utilized for developing various sentiment analysis methodologies are proprietary and have restricted accessibility, posing a significant disadvantage for researchers [11], [27]–[29]. Additionally, the limited access to these datasets makes it challenging for different research groups to compare or replicate studies conducted on the same datasets. This circumstance may lead to potential knowledge loss in terms of developing novel methodologies and comparing existing techniques in the field of sentiment analysis. The availability of openly accessible and standardized datasets can contribute to the advancement of the field by granting researchers access to a broader spectrum of data. In this context, the dataset we have created has been shared openly for accessibility.

Another significant observation in related studies is the variable number of classes and sample sizes in the datasets. This variation makes it difficult to make comparisons and evaluate models effectively [15]–[17]. When the datasets contain different numbers of classes or different sample sizes, it becomes difficult to accurately evaluate the performance of sentiment analysis models. Furthermore, the inequality in class distribution across datasets challenges the process of comparing model performance across different studies. Researchers often face challenges in standardizing evaluation criteria and methodologies due to these differences, hindering the ability to draw meaningful conclusions from comparative analyses. As a result, achieving consensus or establishing best practices in sentiment analysis becomes more difficult in the presence of such dataset variability. We consider it imperative to increase the number of data samples in our experimental study compared to the monitored studies, but we accept that there is a limitation in this regard.

According to the findings obtained from the discussion table, it is evident that datasets obtained from various shopping portals (hotes.com, amazon.com, hepsiburada.com, etc.) were used. This observation indicates that the differences in the native languages of the users providing feedback, quality expectations and many other variable attributes may lead to expected differences in the generalization performance of the proposed models. In fact, reviews on different shopping portals are likely to vary in terms of cultural differences and shopping experiences. Therefore, it is crucial to consider this diversity when evaluating the generalization capabilities of sentiment analysis models based on data from different shopping platforms. These factors may affect the performance of the models and consequently, the variability in the usability of the proposed models across different platforms should be taken into account.

4. Conclusion

E-commerce is revolutionizing sales opportunities by enhancing the shopping experience of customers and expanding the horizons of businesses. Customer reviews have a crucial role in forming the purchasing decisions of potential customers, providing them with real insights into the quality of products and services offered by businesses. In this study, we created an openly accessible dataset of 2000 customer product reviews. For this purpose, we built a custom application called Product Pulse, which is designed to securely store customer reviews indefinitely. In the experimental phase, we performed a comprehensive exploration on the dataset with a total of 36 models and tested each model with varying configurations of hyperparameters based on four different core models. The analysis revealed highly satisfactory and promising results, providing a remarkable classification accuracy of 94.76%. To demonstrate the potential applicability of our proposed model in various online business frameworks and to facilitate its integration with minimal configuration, we created a Review Mate Web API service for model consumption. Furthermore, ReviewMate, a React-based application, was designed to serve as an interface between the API service and end users. We have made the datasets, services and applications from this extensive study available to other researchers through a private GitHub repository.

In our future work, we intend to keep leveraging the Product Pulse tool to collect customer reviews and openly disseminate datasets. In addition, our research efforts will focus on developing new models and tools to deal effectively with the challenges inherent in sentiment analysis based on customer reviews.

Data Availability

The dataset, services, and apps are publicly shared on the GitHub repository. Click here.

References

[1] M. Guha Majumder, S. Dutta Gupta, and J. Paul, “Perceived usefulness of online customer reviews: A review mining approach using machine learning & exploratory data analysis,” *J. Bus. Res.*, vol. 150, pp. 147–164, 2022, doi: https://doi.org/10.1016/j.jbusres.2022.06.012.

[2] S.-H. Chin, C. Lu, P.-T. Ho, Y.-F. Shiao, and T.-J. Wu, “Commodity anti-counterfeiting decision in e-commerce trade based on machine learning and

Internet of Things," *Comput. Stand. Interfaces*, vol. 76, p. 103504, 2021, doi: https://doi.org/10.1016/j.csi.2020.103504.

[3] F. Liu, S. Liu, and G. Jiang, "Consumers' decision-making process in redeeming and sharing behaviors toward app-based mobile coupons in social commerce," *Int. J. Inf. Manage.*, vol. 67, p. 102550, 2022, doi: https://doi.org/10.1016/j.ijinfomgt.2022.102550.

[4] E. M. Mercha and H. Benbrahim, "Machine learning and deep learning for sentiment analysis across languages: A survey," *Neurocomputing*, vol. 531, pp. 195–216, 2023, doi: https://doi.org/10.1016/j.neucom.2023.02.015.

[5] N. Meilatinova, "Social commerce: Factors affecting customer repurchase and word-of-mouth intentions," *Int. J. Inf. Manage.*, vol. 57, p. 102300, 2021, doi: https://doi.org/10.1016/j.ijinfomgt.2020.102300.

[6] X. Xu, R. Fan, D. Wang, Y. Wang, and Y. Wang, "The role of consumer reviews in e-commerce platform credit supervision: A signaling game model based on complex network," *Electron. Commer. Res. Appl.*, vol. 63, p. 101347, 2024, doi: https://doi.org/10.1016/j.elerap.2023.101347.

[7] D. Zhang, L. G. Pee, and L. Cui, "Artificial intelligence in E-commerce fulfillment: A case study of resource orchestration at Alibaba's Smart Warehouse," *Int. J. Inf. Manage.*, vol. 57, p. 102304, 2021, doi: https://doi.org/10.1016/j.ijinfomgt.2020.102304.

[8] O. Araque, J. F. Sánchez-Rada, and C. A. Iglesias, "GSITK: A sentiment analysis framework for agile replication and development," *SoftwareX*, vol. 17, p. 100921, 2022, doi: https://doi.org/10.1016/j.softx.2021.100921.

[9] J. Joung and H. Kim, "Interpretable machine learning-based approach for customer segmentation for new product development from online product reviews," *Int. J. Inf. Manage.*, vol. 70, p. 102641, 2023, doi: https://doi.org/10.1016/j.ijinfomgt.2023.102641.

[10] A. Patel, P. Oza, and S. Agrawal, "Sentiment Analysis of Customer Feedback and Reviews for Airline Services using Language Representation Model," *Procedia Comput. Sci.*, vol. 218, pp. 2459–2467, 2023, doi: https://doi.org/10.1016/j.procs.2023.01.221.

[11] P. Savci and B. Das, "Prediction of the customers' interests using sentiment analysis in e-commerce data for comparison of Arabic, English, and Turkish languages," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 35, no. 3, pp. 227–237, 2023, doi: https://doi.org/10.1016/j.jksuci.2023.02.017.

[12] M. Nilashi, R. A. Abumalloh, S. Samad, M. Alrizq, S. Alyami, and A. Alghamdi, "Analysis of customers' satisfaction with baby products: The moderating role of brand image," *J. Retail. Consum. Serv.*, vol. 73, p. 103334, 2023, doi: https://doi.org/10.1016/j.jretconser.2023.103334.

[13] R. Haque, N. Islam, M. Tasneem, and A. K. Das, "Multi-class sentiment classification on Bengali social media comments using machine learning," *Int. J. Cogn. Comput. Eng.*, vol. 4, pp. 21–35, 2023, doi: https://doi.org/10.1016/j.ijcce.2023.01.001.

[14] H. Karayiğit, Ç. İnan Acı, and A. Akdağlı, "Detecting abusive Instagram comments in Turkish using convolutional Neural network and machine learning methods," *Expert Syst. Appl.*, vol. 174, p. 114802, 2021, doi: https://doi.org/10.1016/j.eswa.2021.114802.

[15] M. Demircan, A. Seller, F. Abut, and M. F. Akay, "Developing Turkish sentiment analysis models using machine learning and e-commerce data," *Int. J. Cogn. Comput. Eng.*, vol. 2, pp. 202–207, 2021, doi: https://doi.org/10.1016/j.ijcce.2021.11.003.

[16] H. Zhao, Z. Liu, X. Yao, and Q. Yang, "A machine learning-based sentiment analysis of online product reviews with a novel term weighting and feature selection approach," *Inf. Process. Manag.*, vol. 58, no. 5, p. 102656, 2021, doi: https://doi.org/10.1016/j.ipm.2021.102656.

[17] G. Cosma and G. Acampora, "A computational intelligence approach to efficiently predicting review ratings in e-commerce," *Appl. Soft Comput.*, vol. 44, pp. 153–162, 2016, doi: https://doi.org/10.1016/j.asoc.2016.02.024.

[18] N. Yücel and Ö. Cömert, "Müşteri Duyarlılığını Keşfetmek İçin Yapay Zeka Destekli Analiz ile Çevrimiçi Ürün İncelemelerinden Anlamlı Bilgiler Elde Etme," *Fırat Üniversitesi Mühendislik Bilim. Derg.*, vol. 35, no. 2, pp. 679–690, 2023, doi: 10.35234/fumbd.1305932.

[19] M. Kilinc and C. Aydin, "Feature selection for Turkish Crowdfunding projects with using filtering and wrapping methods," *Electron. Commer. Res. Appl.*, vol. 62, p. 101340, 2023, doi: https://doi.org/10.1016/j.elerap.2023.101340.

[20] A. R. Webb, *Statistical pattern recognition*. John Wiley & Sons, 2003.

[21] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *J. Mach. Learn. Res.*, vol. 3, no. 3, pp. 1157–1182, 2003, doi: 10.1016/j.aca.2011.07.027.

[22] Z. Cömert and A. F. Kocamaz, "A Study of Artificial Neural Network Training Algorithms for

Classification of Cardiotocography Signals," *Bitlis Eren Univ. J. Sci. Technol.*, vol. 7, no. 2, pp. 93–103, 2017, doi: 10.17678/beuscitech.338085.

[23] Microsoft, "ML.NET An open source and cross-platform machine learning framework," 2024. https://dotnet.microsoft.com/en-us/apps/machinelearning-ai/ml-dotnet

[24] Z. Cömert, A. F. Kocamaz, and V. Subha, "Prognostic model based on image-based time-frequency features and genetic algorithm for fetal hypoxia assessment," *Comput. Biol. Med.*, vol. 99, pp. 85–97, Aug. 2018, doi: 10.1016/J.COMPBIOMED.2018.06.003.

[25] Z. Cömert and Ö. Cömert, "A Study of Technologies Used in Learning Management Systems and Evaluation of New Trend Algorithms," *Bitlis Eren Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, vol. 7. Bitlis Eren Üniversitesi, pp. 286–297, 2018.

[26] A. Sills, M. Ahmed, Dotthatcom.com, F. Boumphrey, and J. Ortiz, "Chapter 6 - Web Development Using XML and ASP.NET," A. Sills, M. Ahmed, Dotthatcom.com, F. Boumphrey, and J. B. T.-X. M. L. N. E. T. D. G. Ortiz, Eds., Rockland: Syngress, 2002, pp. 231–282. doi: https://doi.org/10.1016/B978-192899447-3/50009-3.

[27] P. Rasappan, M. Premkumar, G. Sinha, and K. Chandrasekaran, "Transforming sentiment analysis for e-commerce product reviews: Hybrid deep learning model with an innovative term weighting and feature selection," *Inf. Process. Manag.*, vol. 61, no. 3, p. 103654, 2024, doi: https://doi.org/10.1016/j.ipm.2024.103654.

[28] A. Yucel, A. Dag, A. Oztekin, and M. Carpenter, "A novel text analytic methodology for classification of product and service reviews," *J. Bus. Res.*, vol. 151, pp. 287–297, 2022, doi: https://doi.org/10.1016/j.jbusres.2022.06.062.

[29] P. Vijayaragavan, R. Ponnusamy, and M. Aramudhan, "An optimal support vector machine based classification model for sentimental analysis of online product reviews," *Futur. Gener. Comput. Syst.*, vol. 111, pp. 234–240, 2020, doi: https://doi.org/10.1016/j.future.2020.04.046.

[30] Y. Huang, X. Bai, Q. Liu, H. Peng, Q. Yang, and J. Wang, "Sentence-level sentiment classification based on multi-attention bidirectional gated spiking neural P systems," *Appl. Soft Comput.*, vol. 152, p. 111231, 2024, doi: https://doi.org/10.1016/j.asoc.2024.111231.

[31] L. Gui, Y. Zhou, R. Xu, Y. He, and Q. Lu, "Learning representations from heterogeneous network for sentiment classification of product reviews," *Knowledge-Based Syst.*, vol. 124, pp. 34–45, 2017, doi: https://doi.org/10.1016/j.knosys.2017.02.030.

[32] M. Aydoğan and A. Karci, "Improving the accuracy using pre-trained word embeddings on deep neural networks for Turkish text classification," *Phys. A Stat. Mech. its Appl.*, vol. 541, p. 123288, 2020, doi: https://doi.org/10.1016/j.physa.2019.123288.