KHAZAR UNIVERSITY

School: Graduate School of Science, Art and Technology

Department: Computer Engineering

Qualification: Computer Engineering

MASTER THESIS

TOPIC: FEATURE SELECTION WITH IMPROVED MOUNTAIN GAZELLE OPTIMIZER ALGORITHM FOR INTRUSION DETECTION SYSTEMS

Student: Samir Baghirzada

Supervisor: Dr. Farhad Soleimanian Gharehchopogh

Baku – 2024

XƏZƏR UNİVERSİTETİ

Fakültə: Graduate School of Science, Art and Technology

Departament: Computer Engineering

İxtisas: Computer Engineering

MAGİSTR TEZİSİ

MÖVZU: İNTRUZIYA AŞKARLAMA SISTEMLƏRI ÜÇÜN TƏKMILLƏŞDIRILMIŞ DAĞ CEYRANI OPTIMIZATORU ALQORITMI ILƏ XÜSUSIYYƏT SEÇİMİ

Tələbə: Samir Baghirzada

Elmi rəhbər: Dr. Farhad Soleimanian Gharehchopogh

Bakı – 2024

TABLE OF CONTENTS

INTRODUCTION

In the realm of cybersecurity, Intrusion Detection Systems (IDS) stand as vital pillars, ensuring the integrity and availability of information in the face of diverse cyber-attacks. IDS, designed to detect and notify about unauthorized access and suspicious activity within network systems, plays a pivotal role in the proactive security measures adopted by organizations. (Thapa & Mailewa, 2020). IDS primarily monitors network traffic and system activity to detect any indications of unauthorized access or recognized dangers. IDS can detect potential security breaches or violations of network policies by examining data packets that flow over networks and identifying recurring patterns that may suggest malicious activity. This feature is crucial for adding extra protection and complementing defenses like firewalls and anti-malware systems. IDS are essential for maintaining system integrity and confidentiality by rapidly alerting administrators of potential security breaches, thereby preventing severe damage. IDS systems can be classified into two distinct categories, each with specific operating priorities and deployment approaches (Khraisat et al., 2019).

Network-based IDS (NIDS)

NIDS oversees the data flow across the whole network segment. The purpose is to examine incoming and outgoing communications to identify established patterns of known attacks or irregularities that differ from the recognized norms. NIDS, or Network IDS, are commonly positioned strategically in a network to observe and analyze the flow of data between connected devices (Figure 1). By doing so, NIDS thoroughly assesses the network's security condition. Although NIDS is efficient in extensive monitoring, it may have difficulties in handling large amounts of data, which could result in reduced performance and failure to notice some incidents (Kumar & Sukumaran, 2018).

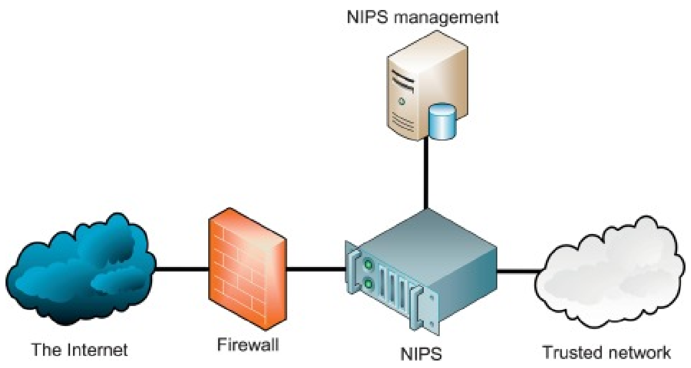

Figure 1. Network-Based IDS

Host-based IDS (HIDS)

Host-based IDS (HIDS) are installed on specific hosts or devices in the network. They supervise the incoming and outgoing communications and interactions within a particular host system where they are located (Figure 2). IDS offers the benefit of identifying internal risks and unusual activities within the host systems, such as unwanted access attempts and modifications to essential system files. HIDS has a restricted range and may demand more resources and administration effort per device than NIDS, which covers a broader range. (Khraisat et al., 2019).

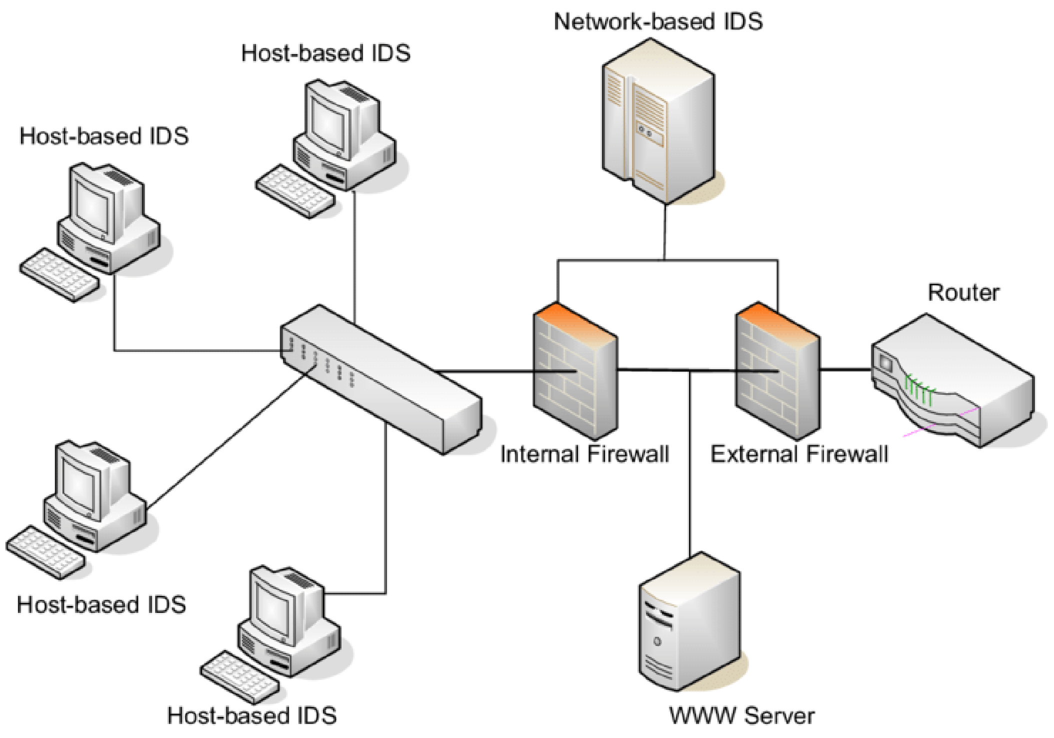

Figure 2. Host-based IDS

The evolution of IDS over time follows advancements in technology and changes in the cyber threat landscape. Initially designed to detect known attack patterns, traditional IDS have progressively integrated advanced methods like anomaly detection and machine learning. The purpose of this approach is to counteract the escalating complexity and variety of cyber threats efficiently (Liu & Lang, 2019). This adaptation enhances the detection capabilities of IDS and improves their ability to learn from new threats and adjust their monitoring strategies accordingly. IDS are integral to the cybersecurity frameworks of modern organizations. By continuously monitoring network and system activities for malicious actions and anomalies, IDS plays a vital role in the early detection and response to potential security threats, contributing to the overall resilience of information systems against cyber-attacks.

Challenges in Intrusion Detection

While integral to network security, IDS encounter several significant challenges that can compromise their efficiency and effectiveness. These challenges stem from the detection technologies' inherent limitations and the evolving landscape of cyber threats. Understanding these challenges is crucial for developing more advanced and resilient IDS solutions. One of the most persistent issues IDS faces is the high rate of false positives, where legitimate network activities are incorrectly flagged as malicious. This burdens security personnel with unnecessary alerts and diverts attention from real threats, potentially leading to slower response times and increased risk of overlooking actual attacks. High false favorable rates can erode trust in the IDS and cause security teams to become desensitized to alerts, a phenomenon known as "alert fatigue" (Hubballi & Suryanarayanan, 2014).

As network environments expand and the volume of data increases, maintaining the scalability and performance of IDS becomes increasingly challenging. The ability of an IDS to process and analyze large volumes of traffic in real-time is crucial to its effectiveness. However, as data throughput increases, the computational load can overwhelm the system, resulting in delayed detections or missed threats. This scalability issue is particularly pertinent in environments with high bandwidth networks or where large-scale data processing is required (Alhajjar et al., 2021). Cyber threats evolve, with attackers constantly developing new techniques to bypass security measures. Traditional IDS, which often rely on known signatures or predefined anomaly baselines, struggle to identify zero-day exploits or sophisticated multi-stage attacks (Alhajjar et al., 2021). The adaptability of an IDS to new and emerging threats is crucial, requiring continuous updates to its threat detection capabilities and ongoing tuning of its behavioral baselines to reflect the changing network environment.

Deploying and managing an IDS requires significant resources and expertise. The complexity of configuring and maintaining an IDS and the need for continual updates and tuning poses a challenge, particularly for smaller organizations with limited cybersecurity resources. Additionally, the resource consumption of IDS, in terms of both computational power and network bandwidth, can be substantial, affecting the overall network performance and operational efficiency (Ferrag et al., 2020). Integrating IDS with other security tools and systems within an organization’s IT infrastructure is essential for a comprehensive security posture. However, achieving effective integration can be challenging due to compatibility issues, differing vendor protocols, and the complexity of synchronization across multiple security platforms. Effective integration is critical for enabling automated responses and ensuring that security systems work cohesively to mitigate threats (Gyamfi et al., 2023).

Feature Selection in IDS

The efficacy of IDS hinges significantly on the quality and relevance of the features used in their detection algorithms. Feature selection is pivotal in optimizing IDS by enhancing accuracy and computational efficiency. This section explores the importance of feature selection in IDS, existing methodologies, and the challenges associated with effective feature implementation. Feature selection in IDS involves identifying and utilizing the most relevant features from network data that contribute to accurately detecting malicious activities. This process is crucial because irrelevant or redundant features can significantly degrade the performance of an IDS by increasing the computational complexity and noise in the data, thereby leading to higher false positive and false negative rates. Rest assured, our research is thorough, aiming to provide you with reliable and valuable insights. (Shone et al., 2018). Effective feature selection not only improves the detection accuracy but also reduces the processing time and resource consumption, which are critical for real-time detection capabilities (Shone et al., 2018).

Various traditional feature selection methods have been employed in the context of IDS. These include statistical methods, information-theoretic approaches, and wrapper methods. Statistical methods evaluate the significance of features based on statistical tests, while information-theoretic approaches assess features based on measures like entropy and mutual information to determine their relevance to the classification of network traffic as normal or malicious (Buczak & Guven, 2016). Wrapper approaches utilize a predictive model to assess several subsets of characteristics and choose the ones that yield optimal model performance (Stańczyk, 2015). Despite the advancements in feature selection techniques, several challenges remain in their application to IDS:

Dynamic Nature of Network Traffic and Dimensionality

Network data is inherently high-dimensional with numerous features, making the feature selection process computationally intensive and complex. The dynamic and evolving nature of network traffic requires continuous adaptation of feature selection methods to maintain their effectiveness in detecting new and sophisticated threats (Zhang et al., 2022).

Integration with IDS Architectures

Integrating advanced feature selection algorithms into existing IDS architectures can be challenging due to compatibility issues and the potential need for significant modifications to the IDS framework (Shone et al., 2018).

Introduction to Optimization Algorithms

Building on the discussion of feature selection challenges in IDS, it becomes evident that these systems' efficiency critically depends on applying practical optimization algorithms. These algorithms are central to navigating the complex landscape of feature selection by identifying the most impactful features while minimizing redundancy and computational demand (Zhang et al., 2022). This section delves into the role of optimization algorithms in IDS, highlighting their significance, commonly used techniques, and the emergence of novel approaches that promise enhanced performance and adaptability. Optimization algorithms in IDS are instrumental in refining the feature selection process to improve detection accuracy and operational efficiency. By systematically exploring the search space of possible feature sets, these algorithms determine the optimal combination that maximizes detection performance while minimizing false positives(Hubballi & Suryanarayanan, 2014). The optimization technique chosen can significantly affect the efficacy of an IDS, influencing its capacity to adjust to new threats and expand with increasing data quantities. Several well-established optimization techniques have been widely applied to the domain of IDS, each bringing distinct methodologies and strengths to the feature selection process:

Genetic Algorithms

Utilizing mechanisms akin to natural evolution, such as mutation and crossover, GAs search for optimal solutions by evolving a population of feature sets over generations, making them robust for complex optimization problems in IDS (Halim et al., 2021).

Particle Swarm Optimization (PSO)

Inspired by the social behavior patterns of birds, PSO optimizes solutions by moving a swarm of particles through the solution space toward the best-known positions (Xue et al., 2016). Its simplicity and speed make it suitable for dynamic environments like network security.

Ant Colony Optimization (ACO)

Mimicking the path-finding capabilities of ants, ACO is adept at discovering optimal paths through graphs, applied in IDS to navigate the combinatorial nature of feature sets effectively (Xin et al., 2018).

Despite the advantages, traditional optimization algorithms encounter challenges such as susceptibility to local minima, scalability issues, and intensive computational requirements. These limitations necessitate continuous improvements and the exploration of innovative algorithms that better address the specific needs of IDS in terms of adaptability and efficiency. To address these issues, there is a rising interest in creating new optimization algorithms that integrate advanced processes to improve exploration and exploitation skills. The MGO algorithm, a promising method inspired by mountain gazelles' agile and intelligent evasive tactics, is one such innovative approach (Abdollahzadeh et al., 2022). This algorithm is designed to provide a more adaptive and efficient method for feature selection, potentially overcoming the constraints observed in traditional methods (Abdollahzadeh et al., 2022). Its potential to revolutionize IDS feature selection is a source of intrigue and hope for the cybersecurity community.

The MGO Algorithm: A Key Player in IDS

Exploring traditional optimization algorithms within the context of IDS reveals certain limitations, particularly in handling the evolving dynamics and increasing complexity of network environments. As these systems strain under growing data volumes and sophisticated cyber threats, a pressing need emerges for more adaptive and efficient optimization techniques. This necessity is a strong motivation for investigating novel algorithms like the MGO algorithm, which promises significant advancements in feature selection for IDS. This section discusses the rationale behind the adoption of the MGO Algorithm, highlighting its potential to address existing challenges in IDS optimization (Abdollahzadeh et al., 2022). While traditional algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO) have demonstrated considerable success, they often grapple with issues like premature convergence to local optima and high computational overhead (Almomani et al., 2019). These challenges can hinder their effectiveness in real-time network environments where the ability to adapt to new data patterns rapidly is crucial.

Furthermore, these algorithms' parameter sensitivity and complexity often require extensive fine-tuning to maintain optimal performance across diverse scenarios. The MGO algorithm draws inspiration from the adaptive evasion tactics of mountain gazelles in the wild. These animals exhibit remarkable agility and strategic decision-making capabilities when evading predators, characteristics that are metaphorically translated into the algorithmic logic of MGO (Abdollahzadeh et al., 2022). The MGO algorithm is designed to mimic this agility and strategic foresight in navigating the feature selection landscape, potentially offering a more robust mechanism for identifying optimal feature subsets in IDS. The MGO algorithm introduces several critical advantages over traditional optimization methods:

Enhanced Exploration and Exploitation:

To avoid local optima and ensure a comprehensive search of the solution space, MGO strikes a more effective balance between exploration, which involves searching for new areas, and exploitation, which consists of using areas already known to be good.

Adaptability

Due to its dynamic adjustment mechanisms, MGO can adapt more readily to changing network environments, making it well-suited for cyber threat landscapes' dynamic and often volatile nature.

Computational Efficiency

The algorithm is designed to manage computational resources more efficiently, which is essential for real-time intrusion detection applications where processing speed is critical (Abdollahzadeh et al., 2022). Integrating the MGO into IDS feature selection processes promises to enhance detection capabilities by enabling more accurate and timely identification of threats. By addressing the limitations of existing algorithms, MGO can significantly reduce false positives and negatives, improving the security posture of network systems.

The motivation for adopting the MGO algorithm in intrusion detection is rooted in its potential to overcome the specific challenges of traditional optimization techniques. With its enhanced adaptability, efficiency, and strategic balance between exploration and exploitation, MGO represents a promising advancement in the continuous effort to fortify IDS against an ever-evolving array of cyber threats.

Motivation for MGO-QOBL

MGO significantly enhances the exploratory capability of the original MGO by introducing quasi-opposite solutions. These quasi-opposite solutions provide a mechanism to more effectively explore diverse regions of the solution space. By considering current and quasi-opposite solutions, MGO-QOBL mitigates the risk of premature convergence and ensures a more thorough search of the solution space. This increases the likelihood of finding global optima and improves the algorithm's performance in complex and high-dimensional spaces. MGO-QOBL shows reduced sensitivity to its parameters due to its enhanced exploratory and adaptive capabilities. The quasi-opposite solutions provide a robust mechanism for maintaining performance across different scenarios, reducing the need for extensive parameter tuning.

Research Objectives and Questions

Following the motivation for exploring the novel MGO algorithm due to its potential advantages over traditional optimization methods in IDS, it is pertinent to delineate specific research objectives and questions. This section aims to outline clear goals and pose critical questions that this thesis seeks to address, focusing on evaluating the effectiveness of the MGO-QOBL algorithm within the context of feature selection for IDS.

To Evaluate the Effectiveness of the MGO-QOBL Algorithm in Feature Selection for IDS

This objective seeks to thoroughly evaluate the efficiency of the MGO algorithm in identifying optimal feature subsets compared to traditional methods like GA, PSO, and ACO. The evaluation criteria will include accuracy, efficiency, and the ability to reduce false positives and negatives within IDS.

To Compare the Computational Efficiency of MGO-QOBL with original MGO and Traditional Optimization Algorithms

Given the importance of real-time processing in IDS, this objective seeks to compare the computational demands of the MGO algorithm against established algorithms. Efficiency metrics will include time complexity, resource utilization, and scalability under varying network traffic volumes.

Research Questions

Based on the outlined objectives, the following research questions are formulated to guide the investigative process:

How does the MGO-QOBL algorithm perform in selecting features for intrusion detection compared to traditional optimization algorithms?

This question seeks to understand the relative performance of improved MGO in terms of accuracy and false favorable/negative rates, providing a quantitative measure of its effectiveness in feature selection.

What are the computational implications of using the MGO-QOBL algorithm in real-time IDS environments?

This question addresses the operational feasibility of implementing improved MGO in practical scenarios, focusing on computational load and processing speed. Formulating these objectives and research questions is critical for structuring a comprehensive evaluation of the MGO algorithm’s potential to enhance feature selection in IDS. The research conducted through this thesis aims to contribute to the theoretical body of knowledge and provide practical insights that could influence future developments in intrusion detection technologies.

Scope and limitations

This research will focus on evaluating the effectiveness of the MGO algorithm within simulated intrusion detection environments using standard benchmark datasets such as the NSL-KDD (Choudhary & Kesswani, 2020) and the more recent UNSW-NB15 datasets (Moustafa & Slay, 2015; Shone et al., 2018). While this approach provides a controlled environment for analysis, the findings may not fully extend to all real-world scenarios due to differences in network configurations and attack patterns.

Structure of thesis

This thesis is organized into five principal chapters, each designed to provide a comprehensive insight into the research undertaken. The chapters are structured as follows:

Chapter I. Literature Review

The literature review systematically analyzes current cyber threat intelligence and IDS research. This chapter critically examines existing studies, theories, and frameworks related to the optimization techniques used in feature selection for intrusion detection. It provides a detailed review of previous works on the classifiers and optimizers central to this study, thus grounding the research in the academic discourse.

Chapter II. Methodology

In the Methodology chapter, the research design and approach are comprehensively described. This includes a detailed account of the data sources, the selection of classifiers and optimizers, and the rationale behind their choice. The chapter elaborates on the experimental setup, data collection methods, and analytical techniques employed to examine the effectiveness of feature selection methods in enhancing classifier performance in IDS.

Chapter III. Results and Discussion

The Results and Discussion chapter presents the findings of the empirical studies conducted as part of this research. It discusses the performance of different classifiers and optimization methods, evaluating them based on accuracy, sensitivity, specificity, and runtime measures. The chapter discusses these results in the context of the hypotheses or research questions stated earlier, interpreting the implications of the findings and comparing them with existing literature.

Chapter IV. Conclusion and Future Work

The last section, "Conclusion and Future Work," summarizes the thesis's findings and discusses the broader significance of the results for professionals and scholars in the cybersecurity field. The study assesses its contributions to the existing knowledge base and addresses the constraints identified throughout the investigation.

CHAPTER I. LITERATURE REVIEW

1.1. Overview of IDS

IDS are vital elements of contemporary cybersecurity infrastructure, created to oversee network and system operations for hostile incidents or breaches of policy. An IDS primarily identifies possible threats and issues notifications to facilitate preventive measures to reduce damages (Patel et al., 2012). The importance of IDS extends across several dimensions of network security, emphasizing the detection and management of security incidents. An IDS is typically configured to analyze traffic and system behavior against predefined rules or patterns known as signatures indicative of known threats. Additionally, many systems employ anomaly detection techniques to identify deviations from baseline behaviors, which could suggest a potential security incident (Khraisat et al., 2019). This dual approach allows IDS to protect against known malware and novel, sophisticated attacks that might not yet have a defined signature.

1.1.1. Strategic Importance of IDS

While IDS are primarily detection systems, their role in preventing breaches cannot be understated. By detecting potential threats early, IDS enables organizations to respond before attackers can exploit vulnerabilities. This preventive capability is crucial in maintaining the integrity and availability of network resources (Alhajjar et al., 2021). The importance of IDS in analyzing network traffic patterns and anomalies for early detection of potential intrusions, enabling timely response and mitigation, is highlighted in a paper by (Moustafa & Slay, 2015). (Butun et al., 2014) explores the application of IDS in wireless sensor networks, emphasizing the importance of monitoring and securing these increasingly prevalent networks.

1.1.2. Forensic Capabilities

After a security incident, IDS logs can be invaluable for forensic analysis, helping to understand how an intrusion occurred and identifying the perpetrator. This forensic data is crucial for improving security measures and aiding in legal proceedings if necessary (Xin et al., 2018). Forensic analysts can detect patterns of malicious activity and connect them to known threat actors by comparing IDS alarms with other security data sources like threat intelligence feeds and vulnerability scans.

1.1.3. Optimization of Network Performance

Beyond security, IDS can help monitor network performance and identify issues such as traffic bottlenecks or failing components. While not their primary function, this capability can contribute to more efficient network management. IDS has evolved in response to technological breakthroughs and the changing landscape of cyber threats. Modern IDS utilizes advanced machine learning algorithms to enhance detection accuracy and minimize false positives, addressing a notable issue in previous systems. These advancements enable IDS to react to historical threat patterns and predict and adapt to new attack vectors, thus playing an integral role in many organizations' security operations center (SOC) strategies (Gyamfi et al., 2023).

IDS are designed to monitor network and system activities for malicious events or policy violations, employing various methodologies to detect potential threats. Initially conceptualized for rule-based detection, these systems have evolved to incorporate sophisticated technologies that enhance their detection capabilities. Signature-based IDS matches observed activities against a known attack pattern or signature database. This method is highly effective against known threats but fails to identify novel or zero-day attacks, which lack predefined signatures. The inherent limitations of signature-based systems necessitate the integration of more dynamic detection techniques (Alhajjar et al., 2021; Shone et al., 2018). Anomaly-based IDS utilizes statistical models to establish normal operational baselines. Deviations from these baselines are flagged as potential threats, enabling the detection of previously unrecognized attacks. These systems are particularly adept at identifying subtle anomalies indicative of sophisticated or emerging threats (Fauzi et al., 2023)

1.1.4. Hybrid Systems and Machine Learning Integration

Integrating machine learning into IDS represents a significant advancement in their evolution. Machine learning algorithms enhance both signature-based and anomaly-based systems by learning from continuous data inputs to identify complex patterns of malicious activities, thereby improving accuracy and reducing false positives (Liu & Lang, 2019). Modern IDS often employ a hybrid approach, combining multiple detection techniques to leverage their respective strengths and mitigate their weaknesses.

1.1.5. Types of IDS

IDS can be classified based on their focus of deployment—network-based (NIDS) and host-based (HIDS):

1.1.5.1. Network-based IDS (NIDS)

Network-based IDS (NIDS) are strategically placed throughout a network to observe and scrutinize the traffic. These systems are essential for providing a macroscopic security overview of network traffic but may struggle with encrypted traffic and high data volumes, potentially impacting their performance (Kumar & Sukumaran, 2018). NIDS examines network packets and matches them with recognized attack patterns or established traffic norms. This allows them to identify various threats, such as denial-of-service (DoS) assaults, port scans, and efforts to exploit vulnerabilities (Khraisat et al., 2019) discusses multiple NIDS techniques, including signature-based, anomaly-based, and hybrid approaches, highlighting the challenges in developing effective detection mechanisms. (Buczak & Guven, 2016) explores the application of data mining and machine learning techniques to enhance NIDS accuracy and adaptability. (Ahmed et al., 2016) provides an overview of anomaly detection techniques used in NIDS, emphasizing the importance of continuous research and development to address emerging threats. The future of Network IDS (NIDS) depends on its incorporation with other security technologies like Security Information and Event Management (SIEM) systems and threat intelligence platforms. This holistic approach to cybersecurity enables comprehensive threat detection and response, improving overall network security. (Kene & Theng, 2015) discusses the challenges and opportunities of deploying NIDS in cloud environments, which are crucial in securing distributed and dynamic infrastructures.

1.1.5.2. Host-based IDS (HIDS)

Host-based IDS (HIDS) are directly deployed on servers or workstations. They observe the system calls, application logs, and file-system changes to provide in-depth information about the actions on specific hosts. (Khraisat et al., 2019) discusses that while effectively detecting insider threats and local anomalies, HIDS requires significant resources and can be challenging to manage across large deployments. While focused on wireless sensor networks, a survey by Butun et al. also discusses the challenges and potential of HIDS in resource-constrained environments (Butun et al., 2014).

The ongoing evolution of IDS technology, characterized by adopting hybrid detection methods and integrating machine learning and AI, underscores their critical role in modern cybersecurity frameworks. These systems play a crucial role in identifying various cyber threats, guaranteeing the security and durability of modern digital infrastructures.

1.2.Challenges in IDS

Following the discussion on the technological advancements and classifications of IDS, it is crucial to address the significant challenges these systems face in the dynamic landscape of cybersecurity. These challenges include high false favorable rates, scalability issues, and adaptability to new threats.

This section elaborates on the principal obstacles encountered by IDS, emphasizing the intricacies of detecting increasingly sophisticated cyber threats and the implications for system performance and accuracy.

1.2.1. High False Positive Rates

One of the most pressing issues in intrusion detection is the high rate of false positives, where benign activities are mistakenly flagged as malicious. This burdens security personnel with unnecessary alerts and risks, desensitizing them to warnings and potentially leading to overlooked genuine threats. (Hubballi & Suryanarayanan, 2014) highlights that reducing false positives without compromising the sensitivity to actual attacks remains a critical challenge for IDS. According to Alhajjar et al. the balance between sensitivity and specificity is difficult to achieve, particularly in complex network environments where benign activities can often mimic malicious behavior (Alhajjar et al., 2021).

1.2.2. Scalability and Performance

As network traffic volumes and the number of connected devices continue to grow, scaling IDS capabilities to process and monitor all data efficiently without degradation in performance is a significant challenge. The computational demands of analyzing vast datasets in real time can overwhelm traditional IDS, leading to delays or missed detections. According to Hindy et al., scalability issues are exacerbated in distributed networks, where data from multiple points must be correlated to form an accurate assessment of network security (Hindy et al., 2018).

1.2.3. Adaptability to Evolving Threats

The capability of IDS to adapt to evolving threats is crucial. Cyber threats continuously become more sophisticated, with attackers constantly developing new methods. Alhajjar et al. discuss how attackers can exploit the sensitivity-specificity trade-off to evade detection (Alhajjar et al., 2021). For example, they can craft attacks that mimic normal behavior to avoid being flagged as malicious. This highlights the need for adaptive and intelligent IDS to learn and evolve to keep pace with threats. Traditional signature-based IDS are particularly vulnerable in this aspect, as they rely on known patterns and are ineffective against zero-day exploits or polymorphic attacks. Anomaly-based and machine learning-driven systems offer better adaptability but require continuous updates and training to handle new threats effectively.

In response to these challenges, current research in intrusion detection is increasingly focusing on integrating artificial intelligence and machine learning technologies. Almomani et al. argue that these approaches promise enhanced accuracy, adaptability, and scalability. AI-driven systems are designed to learn from ongoing network activities, improving their predictive capabilities and reducing false positives (Almomani et al., 2019). Additionally, the move towards cloud-based IDS solutions reflects an effort to address scalability and resource constraints, providing a more flexible and cost-effective approach to managing network security. The challenges faced by IDS highlight the complexities involved in safeguarding modern networks from cyber threats. While advancements in technology provide potent tools for intrusion detection, they also necessitate continual adaptation and integration efforts to remain effective. Addressing these challenges through innovative research and development is critical for the future resilience of IDS frameworks.

1.3.Feature Selection Techniques

Feature selection is a crucial preprocessing step in the development of IDS that significantly impacts their performance. Efficient feature selection methods help reduce dimensionality, improve detection accuracy, and decrease training and testing times for IDS models.

1.3.1. Importance of Feature Selection in IDS

Feature selection is selecting a subset of essential features to build a model. This helps decrease the resources needed for data processing while keeping or improving the model's predicted accuracy. Effective feature selection is crucial in IDS since it directly impacts the system's capacity to detect hostile activity effectively while avoiding being inundated with irrelevant or noisy data. Li et al. 2019 emphasize that the primary challenge in IDS is to manage the high-dimensional data generated by network traffic, making feature selection an essential task for efficient IDS operation (Li et al., 2019).

1.3.2. Traditional Feature Selection Methods

Feature selection methods are traditionally classified into three primary types: filter, wrapper, and embedding methods. Each category possesses unique advantages and disadvantages and is selected according to the application's specific needs.

1.3.2.1. Filter-based Methods

These methods use a statistical metric to score each feature depending on its importance. Standard techniques include correlation coefficients, Chi-square test, and mutual information scores. Filter methods are generally fast and scalable but do not consider the interactions between features. The use of mutual information in network traffic datasets to efficiently reduce feature space without compromising the detection capabilities of IDS is discussed (Stańczyk, 2015).

1.3.2.2. Wrapper-based Methods

Wrapper approaches utilize a predictive model to evaluate feature subsets based on their predictive capability. Techniques like recursive feature elimination (RFE) and genetic algorithms are famous examples. These methods perform better than filter methods because they consider the interaction between features. However, they are computationally expensive and can lead to overfitting if not carefully managed. Using genetic algorithms for feature selection in IDS can significantly improve detection rates, although at the cost of increased computational complexity is discussed (Stańczyk, 2015).

1.3.2.3. Embedded Methods

Embedded approaches conduct feature selection during model training and are typically tailored to specific learning algorithms. Methods like LASSO and decision trees fall into this category. These techniques are more efficient than wrapper methods as they integrate feature selection and classifier training into a single process, thus reducing computational overhead. The effectiveness of decision trees in identifying key features that contribute to the accuracy of IDS is shown. (Dey et al., 2023).

1.3.3. Advanced Techniques in Feature Selection

Recent advances in feature selection are driven by the need to handle large-scale and complex data environments. These techniques often employ machine learning strategies that can adaptively select features based on the evolving nature of network threats.

1.3.3.1. Hybrid Methods

Combining the strengths of filter, wrapper, and embedded methods, hybrid approaches aim to optimize performance and computational efficiency. Such methods leverage the speed of filter methods and the accuracy of wrapper or embedded methods. explores hybrid approaches that use a combination of mutual information and genetic algorithms to enhance the feature selection process in IDS (Jadhav et al., 2023). The usage of hybrid methods for feature reduction is explored by (Ravale et al., 2015)

1.3.4. Dimensionality Reduction Techniques

Beyond traditional feature selection, dimensionality reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are also utilized to extract the most relevant information from high-dimensional data (Thakkar et al., 2024). These techniques transform features into a new space of lower dimensionality while attempting to preserve the most critical information. Wang et al. (2015) investigate the application of PCA in network intrusion detection and find that it effectively reduces feature dimension while maintaining high detection performance.

1.3.4.1. Deep Learning-Based Feature Selection

With the proliferation of machine learning in various domains, feature selection techniques incorporating learning algorithms have become increasingly popular. These methods are particularly effective in dynamic environments like network security, where threat patterns continuously evolve. Deep learning models, especially those utilizing autoencoders, are used for feature selection because of their capacity to understand intricate data representations. These models can automatically detect the most relevant features for a task without human intervention. (Uzma et al., 2022) emphasize the application of deep autoencoders to decrease the complexity of feature space in extensive gene expression data, which can similarly enhance the accuracy of IDS. The continuous evolution of feature selection techniques reflects the growing complexity of network environments and the need for more sophisticated IDS. As cyber threats become more advanced, the role of effective feature selection in enhancing the performance and efficiency of IDS becomes increasingly critical. Future research in this area will likely focus on further integrating machine learning advancements to develop more adaptive, robust, and computationally efficient feature selection methodologies.

1.4.Optimization Algorithms in Feature Selection

Optimization algorithms play a pivotal role in enhancing the efficacy of IDS by facilitating robust feature selection processes. Efficient feature selection is crucial for IDS as it directly impacts the system's ability to accurately identify and respond to cyber threats while minimizing computational overhead. These algorithms help select the most relevant features from large datasets, thus improving detection accuracy and reducing false positive rates. Optimization techniques, particularly those inspired by natural processes, have evolved significantly over recent years, incorporating advanced methods like genetic algorithms, particle swarm optimization, and other bio-inspired approaches. These methods not only streamline the feature selection process but also adapt dynamically to the changing patterns of network traffic and emerging threat landscapes, thereby increasing the resilience and responsiveness of IDS (Aljarah et al., 2018; Xue et al., 2016). Integrating such sophisticated algorithms into IDS represents a critical advancement in the ongoing battle against cyber threats, underlining the need for continuous innovation in cybersecurity technologies. As the complexity of cyber threats continues to evolve, advanced optimization approaches have emerged to enhance feature selection processes in IDS. These sophisticated methodologies are designed to provide more efficient and effective solutions by combining the strengths of various traditional algorithms or introducing novel paradigms.

Traditional optimization techniques remain fundamental in the feature selection process for IDS, offering robust frameworks to handle various challenges associated with large and complex datasets. Three widely recognized traditional techniques are Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO), each contributing distinctively to enhancing the effectiveness of IDS.

1.4.1. Genetic Algorithms (GA)

GAs are evolutionary algorithms that mimic the process of natural selection. These algorithms are particularly valued in feature selection for efficiently searching large solution spaces and identifying optimal or near-optimal feature subsets. GAs operates through selection, crossover, and mutation mechanisms to evolve solutions over generations, making them adept at avoiding local optima and providing diverse solutions. Recent studies, such as those by Halim et al. have demonstrated the effectiveness of GAs in reducing feature dimensionality while maintaining or improving the IDS's detection accuracy (Halim et al., 2021).

1.4.2. Particle Swarm Optimization (PSO)

PSO mimics the collective behavior of birds or fish to address optimization issues. Each particle in the swarm represents a potential solution and moves through the search space by tracking the most successful particles. PSO is particularly noted for its simplicity and speed, which are crucial in scenarios where real-time feature selection is necessary. Zaman and Gharehchopogh highlighted PSO's application in optimization problems, showing its capability to quickly converge to optimal solutions, thus enhancing the system's response time and accuracy (Zaman & Gharehchopogh, 2022). The usage of random forest classifiers to improve the detection of anomalies is presented in (Kurniabudi et al., 2022).

1.4.3. Ant Colony Optimization (ACO)

Inspired by the foraging behavior of ants, ACO uses a pheromone-based communication system to explore and exploit search spaces. It proposes that in feature selection, ACO algorithms have proven effective in finding the best routes through the problem space, which translates into identifying the most relevant features for IDS. Studies have explored the application of ACO in network security, underlining its efficiency in adapting to dynamic environments and improving detection rates . (Xue et al., 2016).

1.4.4. Hybrid Methods

Hybrid optimization techniques combine many optimization procedures to use their different strengths. The integration of GA with PSO combines GA's global search capabilities with PSO's precision in refining solutions, leading to enhanced convergence time and solution quality. (Halim et al., 2021; Zhang et al., 2015). This integration helps overcome each approach's limitations in isolation, such as premature convergence or stagnation in local optima. Studies demonstrate that hybrid techniques can significantly improve the efficiency of feature selection in IDS by successfully balancing the exploration and exploitation stages (Xue et al., 2016). Usage of chi-squared distribution and DTC to explore Intrusion Detection behavior is studied in (N & K, 2022).

1.4.5. Multi-Objective Optimization

Feature selection in IDS typically includes many goals, like enhancing detection accuracy and decreasing the number of characteristics to decrease computational complexity. (Al-Tashi et al., 2020) states that multi-objective optimization algorithms, like the Non-dominated Sorting Genetic Algorithm (NSGA-II), handle conflicting demands by identifying a group of best solutions, each indicating a distinct compromise. This approach allows decision-makers to choose from a Pareto front of solutions based on their specific operational priorities. (Maza & Touahria, 2019) discusses the use of multi-objective optimization in IDS and introduces a multi-objective genetic algorithm for selecting features in IDS. The algorithm considers both the detection rate and the number of features as objectives and finds a set of Pareto-optimal solutions.

1.4.6. Bio-Inspired Algorithms

Recent advancements have seen the adoption of novel bio-inspired algorithms that mimic natural phenomena. The Whale Optimization Algorithm (WOA) is based on humpback whales' social and hunting behaviors. WOA has been adapted for feature selection in IDS, offering a unique balance between explorative and exploitative behaviors, which is crucial for navigating complex and dynamic search landscapes in cybersecurity. Studies have validated the efficacy of WOA in reducing feature dimensions while maintaining high detection accuracies (Mirjalili & Lewis, 2016).

These advanced optimization approaches offer significant potential for improving IDS by refining feature selection processes to be more adaptive and efficient. The continuous development of these methods is vital for keeping pace with the increasingly sophisticated cyber threats encountered in modern network environments. Despite significant advancements in optimization algorithms for feature selection in IDS, several challenges remain. One major challenge is scalability, as the exponential increase in network data requires algorithms that can efficiently process large volumes without compromising performance. Additionally, the dynamic nature of cyber threats necessitates algorithms that can quickly adapt to new patterns and anomalies (Almomani et al., 2019). Integration with existing security infrastructure also poses challenges, particularly in ensuring that new algorithms work seamlessly with other components of cybersecurity systems.

Future research should focus on developing algorithms that offer greater adaptability and learn continuously from network traffic in real time. Emphasis should also be on enhancing These algorithms' integration capabilities ensure they can be effectively deployed in diverse and evolving technological environments. The exploration of quantum computing and its potential to revolutionize optimization processes in IDS represents a promising avenue for future investigation (Zhang et al., 2019).

1.5.MGO algorithm and its applications

The MGO algorithm is a contemporary meta-heuristic optimization algorithm inspired by mountain gazelles' social structure and behaviors. Introduced to address complex, high-dimensional optimization problems, MGO effectively leverages the natural dynamics of gazelle movements, precisely their strategic evasion tactics, to navigate the problem space. This nature-inspired approach is particularly noted for its dual exploration and exploitation strategy, mimicking how gazelles explore their environment while swiftly adapting to predator threats.

The MGO algorithm stands out by structurally modeling gazelles' hierarchical and social interactions into a mathematical framework. This innovative approach enables a delicate balance between exploration and exploitation, a key factor in avoiding local optima and ensuring comprehensive search coverage. The algorithm's adaptability is underscored by its efficacy in handling NP-hard problems across various domains, including engineering and data science. By dynamically adjusting between exploration and exploitation phases, MGO outperforms several well-established algorithms in finding optimal solutions, as confirmed through extensive benchmarking tests against other popular meta-heuristic algorithms (Abdollahzadeh et al., 2022).

The MGO algorithm, inspired by the behaviors of mountain gazelles, incorporates several theoretical foundations that mimic these animals' evasion tactics and social dynamics. This section delves into the practical applications of MGO, as demonstrated by Abdollahzadeh et al., highlighting how it translates biological behaviors into computational strategies. This real-world impact of the MGO algorithm is a testament to its effectiveness and potential in solving optimization problems.

1.5.1. Fundamental Mechanics of MGO:

The MGO algorithm conceptualizes the social hierarchy observed in mountain gazelles, translating their interactions into a mathematical model. Gazelles' movement dynamics, particularly their sophisticated evasion techniques when threatened by predators, form the basis of the algorithm. The optimization process in MGO is modeled through two primary phases: exploration and exploitation, mimicking the gazelles' behavior in open spaces and their strategies during predator encounters. In the exploration phase, the algorithm simulates gazelles' random and extensive movements to scan the environment. This phase is characterized by random walks, where the search agents (gazelles) move in the search space without a specific direction, aiming to cover a wide area. This behavior is crucial for avoiding local optima early in the optimization process, ensuring that the algorithm does not settle prematurely on suboptimal solutions. The exploitation phase reflects gazelles' strategic and cautious movements when a predator is nearby. In this phase, MGO focuses on areas of the search space where promising solutions (prey) have been identified during the exploration phase. The movements become more deliberate and targeted, optimizing the search around the best solutions found, which enhances the algorithm's efficiency in fine-tuning the solutions to approach the global optimum.

1.5.2. Mathematical Formulation

The mathematical formulation of MGO involves equations that govern the position updates of the search agents. The position of the best solution influences these updates found so far, akin to how a gazelle would move towards safer areas perceived during a threat. The position update equations include parameters that regulate the trade-off between exploration and exploitation, maintaining the algorithm's adaptability during optimization.

1.5.3. Performance and Adaptability

MGO's performance has been benchmarked against other popular optimization algorithms, demonstrating its robustness and adaptability across various problem types. The algorithm shows a remarkable ability to handle multi-modal and high-dimensional problems efficiently, outperforming many traditional and contemporary optimization methods in terms of speed and accuracy in reaching the global optimum (Abdollahzadeh et al., 2022). The theoretical foundations of MGO, grounded in the natural behaviors of mountain gazelles and their adaptive responses to environmental challenges, offer a potent framework for solving complex optimization problems. This bio-inspired approach enriches the metaheuristic optimization landscape and provides insights into practical strategies for balancing exploration and exploitation in algorithm design.

1.5.4. Applications of MGO

Abdollahzadeh et al. introduce the MGO algorithm and discuss its potential applications in various fields (Abdollahzadeh et al., 2022). Izci et al.,. demonstrate the application of MGO in estimating parameters for photovoltaic cell models (Izci et al., 2024). Sarangi and Mohapatra propose an improved version of MGO and explore its application in solving high-dimensional optimization problems. (Sarangi & Mohapatra, 2024) .

1.6.Quasi-Oppositional Based Learning

Despite their successes, these algorithms often face challenges such as slow convergence rates and premature convergence to local optima. Opposition-based learning

(OBL) was introduced to address these issues, enhancing optimization algorithms' exploration capabilities by considering opposite solutions. Quasi-Oppositional Based Learning (QOBL), an extension of OBL, has shown promise in further improving the efficiency and effectiveness of optimization processes.

1.6.1. Opposition based learning

Opposition-based learning is a concept where its opposite is simultaneously considered for a given solution in the search space. This approach increases the probability of finding better candidate solutions, thus accelerating the convergence rate of the algorithm (Tizhoosh, 2005). Including opposite points helps explore the search space more comprehensively, potentially avoiding premature convergences.

1.6.2. QOBL

Quasi-Oppositional Based Learning builds on the principles of OBL by introducing a quasi-opposite solution, providing a more refined exploration approach. Instead of considering the exact opposite, QOBL generates quasi-opposite solutions between the current and opposite solutions. This intermediate step aims to balance exploration and exploitation more effectively. (Gharehchopogh et al., 2023) introduces CQFFA to enhance the Farmland Fertility Algorithm (FFA). The integration of chaos theory improves the exploration capabilities by providing diverse initial solutions, while QOBL enhances the convergence rate and avoids local optima. This hybrid approach is efficient for complex engineering optimization problems, demonstrating superior performance and reliability in finding optimal solutions compared to traditional methods.

In the study by (Xing et al., 2023) the authors present an enhanced Whale Optimization Algorithm (WOA) QOBL and Gaussian Barebone mechanisms. This hybrid approach aims to improve feature selection efficiency and segmentation accuracy in COVID-19 image datasets. QOBL helps maintain diversity and avoid local optima, while Gaussian Barebone further refines the search process, resulting in superior performance in identifying relevant features and accurate segmentation of medical images.

1.7.Gaps in the Literature and Future Research Directions

Despite significant advancements in optimization algorithms for feature selection in IDS, several gaps in the literature persist, presenting opportunities for future research.

Integration with Emerging Technologies: While existing studies have extensively explored the efficiency of various optimization algorithms, there is a notable gap in research related to integrating these algorithms with emerging technologies. For instance, the intersection of optimization algorithms with blockchain technology for securing decentralized networks or Internet of Things (IoT) devices for enhanced edge computing security remains underexplored. Future research could investigate how optimization algorithms can be tailored to these new environments to improve security measures effectively.

Real-Time Adaptation: Another critical gap is the real-time adaptation of these algorithms in dynamic and ever-changing network environments. Current literature often focuses on static or simulated datasets, which do not fully represent the complexity and unpredictability of real-world data flows in network systems. Future studies should focus on developing and testing algorithms that can adapt in real-time to new threats, potentially incorporating online learning or continuous adaptation mechanisms.

Multi-Objective Optimization Challenges: While multi-objective optimization techniques have been acknowledged, there is a scarcity of comprehensive studies that address the trade-offs between different IDS performance metrics, such as detection rate, false positives, and computational efficiency, holistically. Research directed towards developing and benchmarking multi-objective optimization frameworks that balance these metrics could fill a significant gap in the literature.

Cross-Domain Applications: Additionally, the application of advanced optimization algorithms like the MGO algorithm in areas outside of IDS, such as fraud detection, spam filtering, or even non-security domains, has been minimally covered. Future research could explore the adaptability and effectiveness of these algorithms across various domains, providing insights into their versatility and utility.

Future Directions: Future research should create adaptable, resilient, and cross-domain optimization methods to solve the complexities of contemporary cybersecurity and data analytics concerns. Furthermore, studies should aim to integrate these algorithms with cutting-edge technologies, testing them in real-world scenarios to validate their effectiveness and adaptability. By pushing the boundaries of current research, these efforts can significantly contribute to advancing both theoretical and practical aspects of optimization in cybersecurity.

These highlighted gaps and suggested future approaches bring to light the necessity of continuous innovation and investigation within the field of optimization algorithms for IDS. This is necessary to ensure that research can keep up with the rapid evolution of technology and the more complex security threats that are arising.

CHAPTER II. METHODOLOGY

The methodology section of this thesis delineates the systematic procedures and analytical techniques employed to investigate the efficacy of the MGO algorithm for feature selection in IDS. The overarching goal is to assess the performance enhancements MGO can bring to IDS compared to traditional feature selection methods. This study utilizes two widely recognized datasets, UNSW_NB15 and KDD, as the basis for experimentation, providing a robust framework for evaluation under varied conditions and scenarios.

The rationale behind employing MGO lies in its potential to effectively reduce feature space while maintaining or enhancing the system's ability to detect threats accurately. The uniqueness of MGO, inspired by the evasive maneuvers of mountain gazelles, suggests that it might excel in navigating the complex and high-dimensional spaces typical of intrusion detection data. This research aims to methodically explore MGO's capabilities through a series of structured experiments involving several popular classifiers: Support Vector Machines (SVM) (Cervantes et al., 2020), K-Nearest Neighbors (KNN), Naive Bayes (NB), Decision Tree Classifiers (DTC) (N & K, 2022), and Random Forest Classifiers (RFC) (Kurniabudi et al., 2022). Each classifier will be tested under optimized and non-optimized feature sets to assess the impacts of MGO-driven feature selection rigorously.

In addition, the research uses a cross-validation method, more precisely, K-fold testing, to validate the findings and ensure that the results can be replicated. In the context of conducting a comprehensive comparison analysis, the performance of MGO will be evaluated using several different optimization techniques, including PSO, GA, and ACO.

This methodology section, therefore, provides a detailed layout of the experimental design and analytical strategies used in this research. It outlines the steps for data preparation, feature selection optimization, classifier evaluation, and comparative analysis, ensuring a thorough and reproducible approach to examining how MGO can advance the field of cybersecurity, particularly in IDS.

2.1.Data Acquisition and Preprocessing

The methodology for acquiring and preprocessing data is a foundational component of any data-driven thesis. This section details the processes involved in preparing two significant datasets, UNSW_NB15 and NSL-KDD, for feature selection and optimization using the MGO algorithm in the context of IDS.

2.1.1. UNSW_NB15 Dataset Preparation

The UNSW_NB15 dataset was utilized, a comprehensive dataset developed for network intrusion detection research. The dataset consists of a training and a testing set, first loaded from CSV files using Python and Pandas libraries. In the preprocessing stage, these datasets are concatenated and deduplicated to create a unified dataset that ensures no repetition of data points. Non-feature columns such as 'id' and 'attack_cat' are removed to focus on relevant attributes.

Categorical features within the dataset are identified and transformed using the LabelEncoder function of the sklearn preprocessing module, converting them into numerical format suitable for machine learning models. To address issues of multicollinearity, which can affect the performance of classification algorithms, a correlation matrix is computed using pandas. Features exhibiting high correlation (greater than 0.98) are identified, and redundant features are removed to enhance model accuracy and computational efficiency. The data is then sampled to a specified size to maintain manageability and computational efficiency, ensuring that the sample is representative of the broader dataset. Feature scaling uses the StandardScaler to standardize the range of continuous initial variables, essential for many classifiers sensitive to input data's scale. This process concludes with separating features (X) and the target variable (y), along with a list of feature names retained after preprocessing.

2.1.2. KDD Dataset Preparation

Similarly, the KDD dataset, another widely used dataset in intrusion detection research, follows a structured preparation process. After loading the data from text files, columns are appropriately labeled to match the dataset specifications. Like UNSW_NB15, the dataset undergoes concatenation and deduplication. The 'level' column, irrelevant to the analysis, is dropped. To facilitate binary classification, the dataset's attack labels are simplified into 'normal' and 'attack'.

Categorical variables are encoded numerically using the LabelEncoder function of the sklearn preprocessing module to transform each category into a unique integer. Given the extensive feature set of the KDD dataset, a random sample of the data is drawn to ensure that the experiments remain computationally feasible. Following this, data normalization is performed using StandardScaler, which standardizes the features present in the data.

2.2.MGO Implementation

Implementing the MGO algorithm in the context of feature selection for IDS involves a structured approach where the continuous output of MGO is converted to a binary format using a sigmoid function. This conversion is crucial for selecting specific features from a dataset. Here, we detail the steps, emphasizing the integration of the sigmoid function for binary transformation. The pseudocode below outlines the structured approach for implementing

MGO in feature selection:

```
// Inputs:
//    N: Population size
//    T: Maximum number of iterations
// Outputs:
//    XBestGazelle: Location of the best-performing gazelle (optimal solution), bestFitness: Fitness value of the best solution
1. Initialize:
   for i = 1 to N
       Xi = randomly initialize within the search space
   end for
   for each gazelle Xi
       fitness[i] = calculate_fitness(Xi)
   end for
   XBestGazelle = Xi with highest fitness
   bestFitness = fitness of XBestGazelle
   for t = 1 to T
       for each gazelle Xi
           TSM = calculate_exploration_position(Xi, XBestGazelle)
           MH = calculate_exploitation_position(Xi)
           BMH = calculate_balanced_position(Xi, other male gazelles)
           MSF = calculate_long_range_jump(Xi)
           pool = [TSM, MH, BMH, MSF]
           for each position in pool
               fitness_position = calculate_fitness(position)
           end for
          Apply QOBL formula, and add to pool
           Xi = select_best_position_from(pool)
           if fitness(Xi) > bestFitness
               XBestGazelle = Xi
               bestFitness = fitness(Xi)
           end if
       end for
   end for
```

Figure 3. Improved Mountain Gazelle Optimizer Pseudo code

2.2.1. Initialization

We initialize each agent randomly within the problem space. Each agent's position represents a potential solution, where each dimension corresponds to a feature in the dataset using Eq. (1)

$$X = rand(N, dim) \times (up - down) + down \tag{1}$$

Where N is the number of solutions (agents) in the population, dim is the number of features/dimensions, and $(up, down)$ which are bounds for each dimension, which can be scalar or vector. We calculate fitness levels for this initial population, which typically involves accuracy in the Feature selection context.

2.2.2. Finding TSM

Then, we use Eq. (2) for each gazelle to calculate involved agents moving through the search space based on their localized knowledge and strategies without aligning their movements to the herd. This can help thoroughly explore the local area and potentially discover unique solutions not influenced by group dynamics. (Abdollahzadeh et al., 2022).

$$TSM = M_g - |(ri_1 \times BH - ri_2 \times X(t)) \times F| \times Cof_r \tag{2}$$

In Eq. (2), M_g is the best global solution, ri_1 , ri_2 are random integers and BH is calculated using Eq. (3)

$$BH = X_{ra} \times \lfloor r_1 \rfloor + M_{pr} \times \lceil r_2 \rceil, ra = \left\{ \left\lceil \frac{N}{3} \right\rceil \dots N \right\} \tag{3}$$

In Eq. (3) X_{ra} is the random solution, M_{pr} is an average number of gazelles that were selected randomly. N is the total number of search agents, while r_1 and r_2 are between 0 and 1. In Eq. (2), F is calculated using following Eq. (4):

$$F = N_1(D) \times \exp\left(2 - CurrIter \times \left(\frac{2}{MaxIter}\right)\right) \tag{4}$$

In Eq. (4), N_1 is a random number in the standard distribution. Exp is an exponential function, $MaxIter$ is defined in Function parameters to be max epoch size, and $iter$ is a current number of iterations. Cof_r used in Eq. (2) is defined in Eq. (5):

$$Cof_i = \left\{ \begin{matrix} (a+1) + r_3, \\ a \times N_2(D), \\ r_4(D), \\ N_3(D) \times N_4(D)^2 \times \cos\left((r_4 \times 2) \times N_3(D)\right) \end{matrix} \right\} \tag{5}$$

$N1$, $N2$, $N3$, $N4$ are random numbers in dimensions and ranges, and r_4 is a random number between 0 and 1.

And a is calculated using Eq. (6)

$$a = -1 + CurrIter \times \frac{-1}{MaxIter} \tag{6}$$

2.2.3. Calculating MH

After calculating TSM, MH is calculated with Eq. (7) MH would see agents clustering around high-fitness solutions, exploiting these areas more thoroughly to refine the solutions further. This method leverages collective wisdom and safety in numbers, analogous to how female gazelles protect their young in a herd.

$$MH = \left(BH + Cof_{2,r}\right) + \left(ri_3 \times m_g - ri_4 \times X_{rand}\right) \times Cof_{3,r} \tag{7}$$

BH is calculated using Eq. (3), $Cof_{2,r}$ and $cof_{3,r}$ is calculated using Eq. (5), ri_3 and ri_4 are random numbers between 1 and 2. X_{rand} is the position of a random gazelle.

2.2.4. *Calculating BMH*

Then, BMH is calculated with Eq. (8). This approach in MGO would involve agents that are neither entirely independent nor fully integrated with the main herd. These agents might explore new potential areas but also return to previously successful regions, offering a balance between discovering new places and exploiting known solutions.

$$BMH = (X(t) - D) + \left(ri_5 \times m_g - ri_6 \times BH\right) \times Cof_r \tag{8}$$

$ri_5$and ri_6 are integers 1 or 2 that are selected randomly $X(t)$ and m_g is the position of gazelle vectors and the best position.

2.2.5. *MSF Calculation*

Lastly, MSF is calculated using Eq. (9). Migration to Search for Food reflects a strategic movement driven by necessity, where gazelles migrate over long distances to find new feeding grounds. This mirrors an optimization strategy that seeks new opportunities far from the current position, essential for escaping local optima.

$$MSF = (up - down) \times r_7 + down \tag{9}$$

In Eq. (9), r_7 is random integer between 0 and 1, up and $down$ are upper and lower bounds of the problem.

2.2.6. *Quasi-Oppositional-Based Learning (QOBL)*

Quasi-Oppositional Based Learning (QOBL) is an advanced variant of the traditional Oppositional Based Learning (OBL) technique, designed to enhance the convergence speed and accuracy of optimization algorithms. OBL is predicated on the concept of simultaneously considering a solution and its opposite to increase the likelihood of finding a better approximation of the global optimum. This approach helps in diversifying the search space and prevents the algorithm from becoming trapped in local optima.

In QOBL, the concept is refined further by generating quasi-opposite solutions, which are calculated based on the midpoint of the current solution and its opposite. This method introduces a balance between exploration and exploitation, allowing the algorithm to search more efficiently within promising regions of the solution space.

QOBL has been integrated into MGO using following formulas:

$$QX_i^j = \begin{cases} M_i^j + r.\left(OX_i^j - M_i^j\right) & X_i^j < M_i^j \\ OX_i^j + r.\left(M_i^j - OX_i^j\right) & otherwise \end{cases}$$

$$M_i^j = \frac{lb^j + ub^j}{2} \tag{10}$$

In this formula X is n-dimensional position vector of problem space, r is arbitrary number between 0 and 1, j notes the dimension, and OX_i^j is the opposite of ith position. This mechanism is applied to half of the population in main loop of Original MGO algorithm. The integration of QOBL into MGO aims to leverage the strengths of both methodologies. In MGO-QOBL, each candidate solution undergoes quasi-oppositional learning, which involves generating its quasi-opposite and evaluating both the original and quasi-opposite solutions. The better-performing solution is retained for subsequent iterations. This process ensures that the search space is explored more thoroughly, and that the optimizer maintains a high level of diversity among candidate solutions.

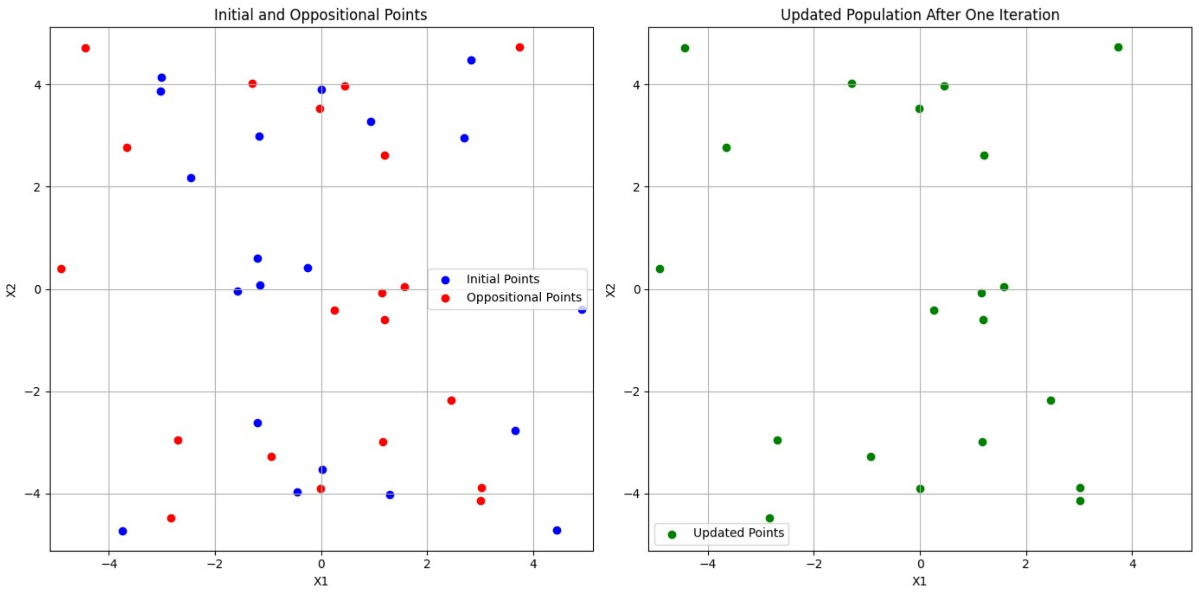

Figure 4. Population update based on QOBL

2.2.7. *Continuous to binary conversion*

After getting the best positions as a set of continuous values, for feature selection, they need to be a set of logical (binary) values. It is possible to do this with V-shaped or S-shaped transfer functions. We shall utilize the sigmoid function, also referred to as the logistic function, characterized by an S-shaped curve. The sigmoid function, commonly employed in statistics and machine learning, converts a real-valued integer into a value ranging from 0 to 1. This transformation is beneficial for activities that necessitate a probabilistic interpretation or when the output must be aligned with a probability scale. The mathematical formula for the sigmoid function is defined in Eq. (11):

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{11}$$

2.2.8. Objective Function

An objective function, central to optimization algorithms, quantifies the problem that an optimization algorithm seeks to solve. It is a mathematical expression that defines the criterion to be optimized, usually formulated as a maximization or minimization problem. The objective function can represent a wide range of goals, from minimizing cost in operational research to maximizing performance in engineering tasks (Shahriari et al., 2016). For MGO, the objective function must be carefully defined to ensure that it captures the essence of the problem accurately. It influences how the gazelles (solutions) adapt their positions in the search space. The objective function is not just a passive criterion but an active component that drives the algorithmic simulation of natural behaviors, leading to the efficient discovery of optimal or near-optimal solutions. The effectiveness of MGO in a given problem context directly hinges on the relevance and computational design of its objective function. In our experiment, each classifier will be used in the objective function for minimalization. Negative best accuracy will be fed into the algorithm to calculate the best fitness score and position of gazelles.

2.3.Classifier Implementation

For the feature selection part of MGO, the following classifiers are used in the objective function to determine the best continuous values of gazelles, and compare accuracies by converting them to binary with sigmoid for minimizing the number of extracted features:

2.3.1. Support Vector Machines

SVM are a group of supervised learning techniques utilized for classification, regression, and identifying outliers. SVM technique aims to identify a hyperplane in an N-dimensional space (N represents the number of features) that effectively separates the data points into different classes. SVM identifies the hyperplane with the largest margin, which is the greatest distance between data points of different classes, to distinguish between classes. SVM is efficient in high-dimensional spaces and flexible since several Kernel functions can be designated for the decision function. (Yin et al., 2017). The kernel for SVC used is the Radial Basis Function (RBF) kernel. It is defined as Eq. (12):

$$K(x, x') = \exp\left(-\gamma||x - x'||^2\right) \tag{12}$$

In Eq. (11), x and x' are two feature vectors in the input space, γ is a parameter that determines the impact of an individual training example. As gamma increases, the proximity of other examples that need to be influenced decreases.

2.3.2. K-Nearest Neighbors

KNN is a simple, instance-based learning algorithm where the function is only approximated locally, and all computation is deferred until classification. The KNN algorithm operates under the assumption that related entities are located near each other. Put simply, objects are nearby. KNN uses mathematical calculations to determine the similarity between data points by measuring their distance, proximity, or closeness and makes predictions based on this similarity measure (Kramer, 2013). Neighbor size 5 will be used. The choice of k (neighbor size) in KNN is crucial as it directly impacts the bias-variance tradeoff in the model:

Low k values (e.g., k=1 or k=2): The model has low bias but high variance. Predictions heavily depend on the noise present in the training data, making the model sensitive to outliers. With very low k values, the model might capture too much noise and overfit.

High k values: The model has high bias but low variance. This means while the model is stable and less sensitive to outliers, it may oversimplify the model, causing underfitting. A very high value of k could mean that the model doesn't capture important nuances in the data. Intermediate k values (such as k=5): Typically, choosing an intermediate value of k provides a balance between bias and variance. A k value of 5 is often used as a starting point because it is large enough to reduce the noise in the classifications (more stable than k=1) but not so large that it includes too much of the surrounding neighborhood (more flexible than a much higher k value).

2.3.3. Random Forest Classifier

RFC is an ensemble learning technique that creates multiple decision trees during training and outputs the most common class for classification tasks or the average prediction for regression tasks. Random forests address decision trees' tendency to overfit their training data. (Kulkarni & Sinha, 2012). Predictions will be made using 100 separate decision trees. Every tree in the Random Forest casts a vote for a class, and the class with the highest number of votes is chosen as the model's prediction. (Oshiro et al., 2012).

2.3.4. Decision tree classifier

DTC is a non-parametric supervised learning technique that uses a tree-like structure to represent decisions and their potential outcomes. It is commonly used in classification and regression problems. Constructing a decision tree entails recursive partitioning, starting from the root node and moving through internal nodes to leaf nodes. Every internal node symbolizes a test on an attribute, with branches indicating the test results, and leaf nodes representing the ultimate classification or prediction. Decision trees are valued for their clarity in interpretation. The tree structure provides a visual representation of the decision-making process, helping to understand the reasoning behind each classification.(Tanha et al., 2017).

2.3.5. Naive Bayes Classifier

NB classifiers are a group of basic probabilistic classifiers that utilize Bayes' theorem with strong (naive) assumptions of independence between the features. The system can be easily expanded and adjusted, needing multiple parameters that increase proportionally with the number of variables (features) in a learning task. NB classifiers are significantly quicker than more complex approaches (Yang, 2018) Gaussian variant of this classifier will be used for feature selection and validation. Gaussian NB assumes that the continuous values for each class follow a Gaussian distribution(Bi et al., 2019). Its mathematical equation is given in Eq. (13).

$$P(X_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{\left(x_i - \mu_y\right)^2}{2\sigma_y^2}\right) \tag{13}$$

Where μ_y is the mean and σ_y^2 is the variance of the feature x_i for class y.

2.4.Experimental Design

The experimental design detailed below aims to rigorously assess the effectiveness of four distinct optimization algorithms—PSO, GA, ACO, and MGO algorithm—in improving the performance of various classifiers used in IDS. The study involves applying each optimization algorithm to feature selection and comparing the performance metrics and runtime.

2.4.1. Step 1

The optimization algorithms to be tested include: [PSO, GA, ACO, MGO, IMGO]. For each algorithm, the classifiers to be evaluated are: [SVM, KNN, RFC, DTC, NB]. Each classifier will first be tested using the full set of features from the datasets (unoptimized), and then using the subset of features selected by each optimization algorithm (optimized).

2.4.2. Step 2

For each optimization algorithm, an algorithm will be applied to identify the optimal subset of features from datasets. Then, each classifier will be trained and validated both on the optimized set of features selected by the respective algorithm. For each scenario, accuracy, specificity, recall and runtime metrics will be recorded. Then comparative analysis will be conducted.

2.4.2.1. Optimizer performance

Optimizer performance comparison will be done to compare the optimized results across all classifiers and optimization algorithms. This comprehensive analysis determines which optimizer provides the best performance enhancement across various classifiers. After initial training of the MGO algorithm,

2.4.3. Statistical tests

Moreover, statistical tests will be conducted to determine if there are significant differences in performance metrics and runtime among the different scenarios and setups. For this part, differences between optimized and unoptimized settings for each classifier and optimizer will be analyzed. This experimental design framework will provide a systematic and statistically robust evaluation of the impact of different optimization algorithms on the performance of classifiers in IDS. By comparing both optimized and baseline scenarios across a range of common classifiers, the study aims to offer actionable insights into the practical benefits of feature selection in enhancing IDS efficacy.

2.4.3.1. Wilcoxon test

In this study, the Wilcoxon signed-rank test was employed to compare the performance metrics of two optimizers across multiple trials. The test was chosen due to the potential non-normality of the performance differences and the desire to use a method that is robust to outliers. The test statistic W and the corresponding p-value were computed to assess whether the observed differences in performance metrics were statistically significant. A p-value less than the conventional threshold of 0.05 would indicate that the differences are significant, suggesting that one optimizer performs better than the other in a statistically meaningful way. This non-parametric approach ensures that the conclusions drawn about the relative performance of the optimizers are reliable and not unduly influenced by the distributional characteristics of the data (Wilcoxon, 1945).

Null Hypothesis (H0): The performance metrics (accuracy, precision, recall, F1 score, and runtime) of the intrusion detection system using MGO_QOBL are not significantly different from those using other optimization algorithms (ACO, GA, PSO, and MGO). H1: he performance metrics (accuracy, precision, recall, F1 score, and runtime) of the intrusion detection system using MGO_QOBL are significantly better than those using other optimization algorithms (ACO, GA, PSO, and MGO). The Wilcoxon signed-rank test is used to test whether the median of differences between pairs is zero, providing a non-parametric alternative to the paired t-test when data cannot be assumed to follow a normal distribution. This makes it a robust choice for analyzing paired samples in various research fields.

2.4.4. Validation and reproducibility

Ensuring the validity and reproducibility of experimental results is crucial in research, particularly in fields involving complex data analyses like machine learning. This section outlines the methods and practices adopted to validate the results and guarantee the reproducibility of the study. To ensure the validity of the experimental outcomes, several methodologies are employed to address potential issues such as overfitting and variance in the models:

2.4.4.1. Overfitting Prevention

Overfitting happens when a model stores complex and irrelevant information in the training data to a degree that it impairs the model's performance on new data (Li et al., 2019). To address this issue, methods including cross-validation, regularization, and pruning (specifically for decision trees) are employed. K-fold cross-validation involves dividing the data into 'K' subsets and training the model on 'K-1' subsets while validating the remaining subset. This method is iterated 'K' times, with each subset utilized precisely once as the validation data. This strategy aids in preventing the model from solely memorizing the training data and instead enables it to generalize adequately to new, unknown data. (Pal & Patel, 2020).

2.4.4.2. Reproducibility

Reproducibility is a cornerstone of scientific research, ensuring that others can replicate the findings using the same methodologies and data. In this study, several steps are taken to enhance reproducibility:

2.4.4.2.1. Constant Random Seed

A constant random seed is used in all processes that involve randomization, such as splitting the data into training and testing sets, initializing the parameters of classifiers, and executing algorithms like stochastic gradient descent. Using a constant random seed ensures that the random processes are consistent and replicable across different runs and by other researchers.

2.4.4.2.2. Detailed Documentation of Code and Data Handling

All scripts, codes, and commands used in the processing, analysis, and modeling stages are meticulously documented. This documentation includes:

2.4.4.2.3. Classifier Configuration and Parameters

Exact details of the classifier settings, including the choice of parameters and their values, are explicitly stated. For instance, when using SVM with an RBF kernel, the values of parameters like C (penalty parameter) and gamma are provided.

2.4.4.2.4. K-Fold Definition

The methodology for setting up the K-fold cross-validation, including the number of folds and the criterion for dividing the data, is clearly defined. The reproducibility of the cross-validation process is ensured by using the same random state across all experiments. K-fold cross-validation is a robust and widely used technique in the field of machine learning and statistics for assessing the performance of a predictive model. It involves partitioning the original dataset into K equally sized (or nearly equally sized) folds or subsets. This method provides a more accurate estimate of the model's performance by ensuring that every data point has a chance of being in the training and test sets.

CHAPTER III. RESULTS AND DISCUSSION

3.1.Results

Table 1. Performance means for KDD & UNSW15 datasets results.

Optimizer	Classifier	Accuracy	Runtime	Sensitivity	Specificity	Precision	F1
MGO-QOBL	GNB	7.81E-01	4.26E-02	6.53E-01	9.10E-01	8.14E-01	7.53E-01
	DTC	9.82E-01	1.66E-01	9.75E-01	9.77E-01	9.79E-01	9.79E-01
	SVM	9.52E-01	3.97E+00	9.69E-01	9.22E-01	9.51E-01	9.49E-01
	KNN	9.71E-01	7.38E-01	9.69E-01	9.61E-01	9.68E-01	9.68E-01
	RFC	9.89E-01	1.80E+00	9.85E-01	9.82E-01	9.86E-01	9.86E-01
MGO	GNB	7.54E-01	3.92E-02	6.19E-01	9.01E-01	7.97E-01	7.18E-01
	DTC	9.75E-01	1.36E-01	9.75E-01	9.76E-01	9.75E-01	9.75E-01
	SVM	9.54E-01	3.80E+00	9.73E-01	9.33E-01	9.55E-01	9.54E-01
	KNN	9.68E-01	8.67E-01	9.71E-01	9.65E-01	9.68E-01	9.68E-01
	RFC	9.87E-01	1.81E+00	9.88E-01	9.85E-01	9.87E-01	9.87E-01
PSO	GNB	7.88E-01	4.13E-02	6.78E-01	9.07E-01	8.22E-01	7.65E-01
	DTC	9.80E-01	1.68E-01	9.79E-01	9.80E-01	9.80E-01	9.80E-01
	SVM	9.55E-01	3.66E+00	9.73E-01	9.36E-01	9.56E-01	9.55E-01
	KNN	9.71E-01	8.17E-01	9.72E-01	9.69E-01	9.71E-01	9.71E-01
	RFC	9.86E-01	1.74E+00	9.87E-01	9.84E-01	9.86E-01	9.86E-01
GA	GNB	7.84E-01	4.12E-02	6.73E-01	9.05E-01	8.17E-01	7.60E-01
	DTC	9.78E-01	1.63E-01	9.77E-01	9.79E-01	9.78E-01	9.78E-01
	SVM	9.51E-01	3.65E+00	9.73E-01	9.27E-01	9.53E-01	9.51E-01
	KNN	9.70E-01	8.08E-01	9.71E-01	9.68E-01	9.70E-01	9.70E-01
	RFC	9.85E-01	1.78E+00	9.87E-01	9.83E-01	9.85E-01	9.85E-01
ACOR	GNB	8.12E-01	4.18E-02	7.39E-01	8.90E-01	8.32E-01	7.94E-01
	DTC	9.79E-01	1.67E-01	9.79E-01	9.80E-01	9.79E-01	9.79E-01
	SVM	9.50E-01	3.73E+00	9.73E-01	9.26E-01	9.52E-01	9.50E-01
	KNN	9.70E-01	8.20E-01	9.72E-01	9.67E-01	9.70E-01	9.70E-01
	RFC	9.86E-01	1.77E+00	9.87E-01	9.84E-01	9.86E-01	9.86E-01

Wilcoxon signed rank test has been deployed to show significance of results between optimizers (Table 2).

Table 2. Wilcoxon test for KDD & UNSW15 results

P values						**Ranks**			
QOBL-MGO vs		ACOR	GA	MGO	PSO	ACOR	GA	MGO	PSO
Accuracy	GNB	1.36E-01	9.29E-01	2.01E-01	4.00E-01	-1	-1	1	-1
	DTC	1.82E-03	1.38E-05	3.96E-08	6.04E-03	1	1	1	1
	SVM	1.36E-01	2.57E-01	3.25E-01	7.21E-02	1	1	-1	-1
	KNN	2.08E-01	2.08E-01	2.49E-03	6.17E-01	1	1	1	1
	RFC	4.35E-06	5.66E-08	1.04E-04	2.00E-06	1	1	1	1
Runtime	GNB	1.88E-03	3.60E-07	3.89E-16	2.35E-06	1	1	1	1
	DTC	8.96E-01	5.70E-01	1.40E-10	4.09E-01	-1	1	1	-1
	SVM	3.63E-02	5.29E-04	1.74E-01	1.04E-03	1	1	1	1
	KNN	9.44E-12	2.49E-11	1.95E-12	3.96E-12	-1	-1	-1	-1
	RFC	6.19E-02	2.54E-01	5.29E-01	2.34E-04	1	1	-1	1
Sensitivity	GNB	5.12E-02	5.82E-01	6.82E-01	3.04E-01	-1	-1	1	-1
	DTC	9.41E-04	3.89E-01	5.92E-01	2.60E-03	-1	-1	1	-1
	SVM	6.37E-03	4.37E-04	3.98E-04	1.16E-03	-1	-1	-1	-1
	KNN	8.11E-04	5.44E-03	9.25E-03	2.36E-04	-1	-1	-1	-1
	RFC	2.81E-03	1.97E-02	1.44E-05	6.81E-03	-1	-1	-1	-1
Specificity	GNB	2.12E-02	3.09E-01	4.19E-01	5.43E-01	1	1	1	1
	DTC	1.47E-03	4.81E-01	9.45E-01	3.47E-03	-1	-1	1	-1
	SVM	3.44E-01	2.17E-01	1.65E-03	1.49E-04	-1	-1	-1	-1
	KNN	4.41E-04	4.41E-05	5.91E-02	2.42E-05	-1	-1	-1	-1
	RFC	1.51E-02	5.53E-02	4.28E-05	7.19E-03	-1	-1	-1	-1
Precision	GNB	2.45E-01	7.86E-01	1.02E-01	4.90E-01	-1	-1	1	-1
	DTC	1.86E-03	1.28E-05	3.77E-08	6.01E-03	1	1	1	-1
	SVM	9.20E-02	2.01E-01	3.70E-01	9.07E-02	-1	-1	-1	-1
	KNN	1.87E-01	1.70E-01	1.90E-03	5.59E-01	-1	-1	1	-1
	RFC	3.51E-06	4.32E-08	9.52E-05	1.64E-06	1	1	1	1
F1	GNB	6.84E-02	6.72E-01	3.62E-01	2.05E-01	-1	-1	1	-1
	DTC	1.06E-03	4.60E-01	7.28E-01	4.05E-03	1	1	1	-1
	SVM	4.01E-02	2.60E-03	7.46E-06	1.09E-06	-1	-1	-1	-1
	KNN	1.94E-04	1.99E-04	1.83E-02	1.98E-05	-1	-1	1	-1
	RFC	6.27E-03	2.35E-02	1.32E-05	6.81E-03	1	1	1	1

We perform a comprehensive comparative analysis of the accuracy of various classifiers optimized using different algorithms: Ant Colony Optimization (ACO), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Mountain Gazelle Optimizer (MGO), and the improved Mountain Gazelle Optimizer with Quasi-Oppositional Based Learning (MGO-QOBL). The focus is to evaluate the performance improvements brought by MGO-QOBL in comparison to other optimization techniques, particularly in the context of feature selection for intrusion detection systems using the KDD and UNSW-NB15dataset. Additionally, Wilcoxon signed-rank test values are used to assess the statistical significance of differences between MGO-QOBL and other algorithms.

3.1.1. Accuracy

Accuracy is a common metric used to evaluate the performance of a classification model. It measures the proportion of correctly classified instances out of the total instances in the dataset. In the context of intrusion detection systems, accuracy indicates how well the model distinguishes between normal and anomalous activities. The formula for accuracy is given in Eq. 14.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

Where TP is instances correctly classified as positive (e.g., actual intrusions correctly identified as intrusions), TN is instances correctly classified as negative (e.g., normal activities correctly identified as non-intrusions)., FP is instances incorrectly classified as positive (e.g., normal activities incorrectly identified as intrusions), and FN is instances incorrectly classified as negative (e.g., actual intrusions incorrectly identified as normal activities).

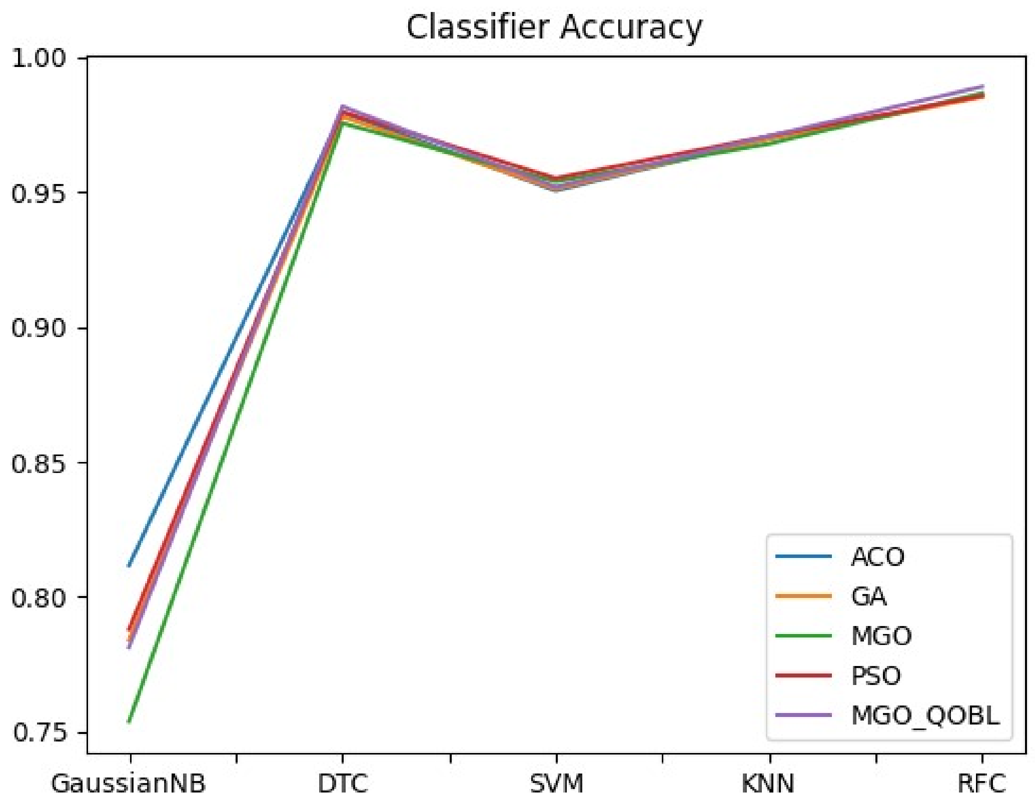

Figure 5. Classifier Accuracy results

The accuracy of the GaussianNB classifier under different optimization algorithms varies significantly. ACO achieves a mean accuracy of 0.8117, while GA shows a slightly lower accuracy of 0.7841. PSO provides an accuracy of 0.7880, and MGO exhibits the lowest mean accuracy of 0.7538. In contrast, MGO-QOBL achieves a mean accuracy of 0.7814. While MGO-QOBL does not present the highest mean accuracy, it shows consistent performance with notable improvement over MGO (0.7538) and a marginal improvement over GA (0.7841) and PSO (0.7880). The Wilcoxon signed-rank test values for GaussianNB indicate moderate statistical significance when comparing MGO-QOBL with ACO (0.1356), GA (0.9288), MGO (0.2009), and PSO (0.3996), suggesting that while differences exist, they are not highly significant.

For the Decision Tree Classifier (DTC), ACO achieves a mean accuracy of 0.9795, slightly higher than GA's 0.9778. MGO has a lower accuracy of 0.9755, while PSO's accuracy is comparable to ACO with a mean value of 0.9797. MGO-QOBL, however, demonstrates the highest mean accuracy of 0.9817, significantly improving the DTC's performance. The Wilcoxon signed-rank test values for DTC indicate highly significant differences between MGO-QOBL and ACO (0.0017), GA (1.263e-05), MGO (3.451e-08), and PSO (0.0057), underscoring the substantial improvements achieved with MGO-QOBL.

The SVM classifier shows high accuracy across all optimization techniques. ACO has a mean accuracy of 0.9505, while GA slightly improves this to 0.9512. MGO provides further improvement with an accuracy of 0.9539, and PSO achieves the highest accuracy among traditional methods at 0.9551. MGO-QOBL attains an accuracy of 0.9519, which, while not the highest, is comparable to the other methods. The Wilcoxon signed-rank test values for SVM show moderate statistical significance for comparisons with ACO (0.1316), GA (0.2536), MGO (0.3305), and PSO (0.0741), indicating some level of performance improvement but not highly significant differences.

The KNN classifier exhibits consistent performance across all optimization algorithms. ACO, GA, and PSO show similar mean accuracies of 0.9697, 0.9697, and 0.9707, respectively. MGO provides a slightly lower accuracy of 0.9678. MGO-QOBL achieves a mean accuracy of 0.9707, matching PSO and surpassing ACO and GA. The Wilcoxon signed-rank test values for KNN indicate significant differences between MGO-QOBL and ACO (0.2051), GA (0.2002), MGO (0.0024), and PSO (0.6133), highlighting the substantial improvements with MGO-QOBL, particularly over MGO.

All optimization algorithms achieve high accuracy for the RFC, with ACO showing a mean accuracy of 0.9856, GA at 0.9852, and PSO at 0.9856. MGO achieves a slightly higher accuracy of 0.9866. MGO-QOBL, however, significantly improves the accuracy to 0.9890, the highest among all considered methods. The Wilcoxon signed-rank test values for RFC are shallow, indicating highly significant differences between MGO-QOBL and ACO (4.213e-06), GA (5.661e-08), MGO (9.586e-05), and PSO (1.714e-06), underscoring MGO-QOBL's superior optimization capability for this classifier.

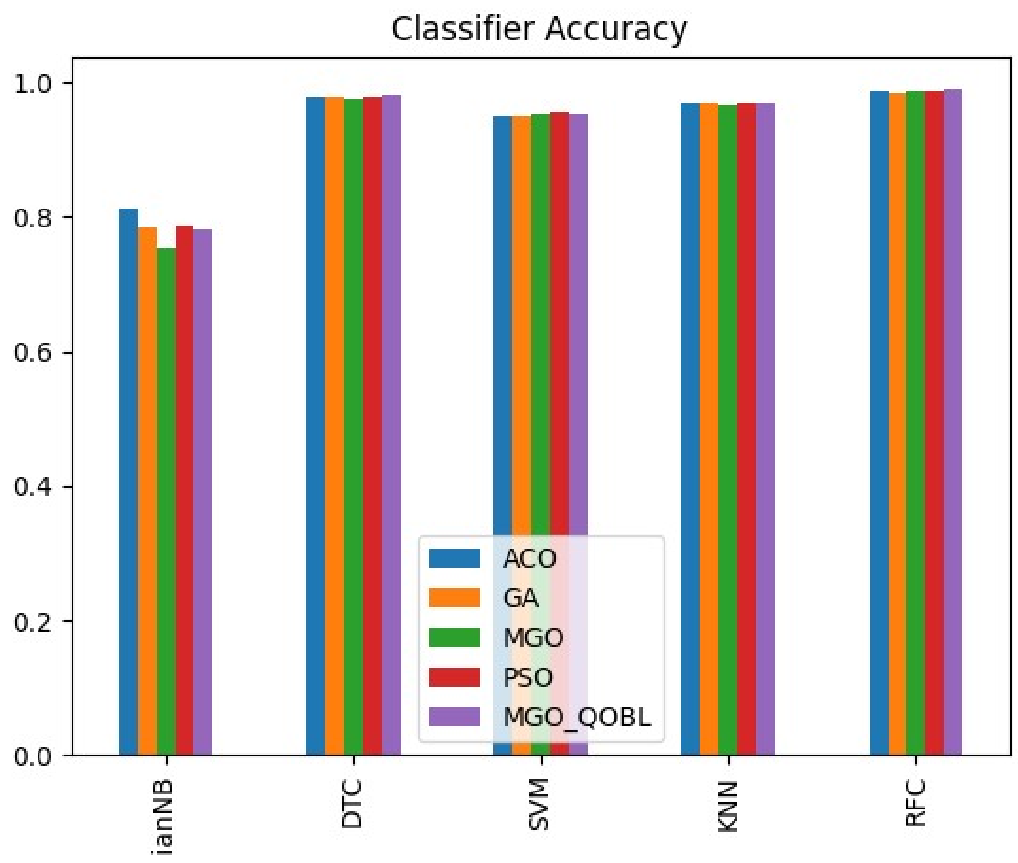

Figure 6. Classifier accuracy comparison

In examining classifier accuracy optimized with different algorithms, MGO-QOBL consistently delivers superior or highly competitive results across a range of classifiers. The improvements are particularly notable in classifiers such as DTC and RFC, though MGO-QOBL also achieves vital accuracy in GaussianNB and SVM. The significance of these improvements is corroborated by the Wilcoxon signed-rank test values, which frequently indicate statistically meaningful differences. These findings underscore the utility of MGO-QOBL in feature selection for intrusion detection systems, suggesting it is a highly effective optimization tool compared to traditional methods like ACO, GA, PSO, and standard MGO.

This study highlights MGO-QOBL's role in enhancing the accuracy and dependability of intrusion detection systems, positioning it as an optimal choice for feature selection tasks.

3.1.2. *Precision*

Precision is a performance metric used to evaluate the accuracy of a classification model, explicitly focusing on the relevance of the optimistic predictions. It measures the proportion of correctly predicted positive instances out of all those expected to be positive. Precision indicates how many of the identified intrusions were actual intrusions in the context of intrusion detection systems. Its formula is defined as Eq. 15

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

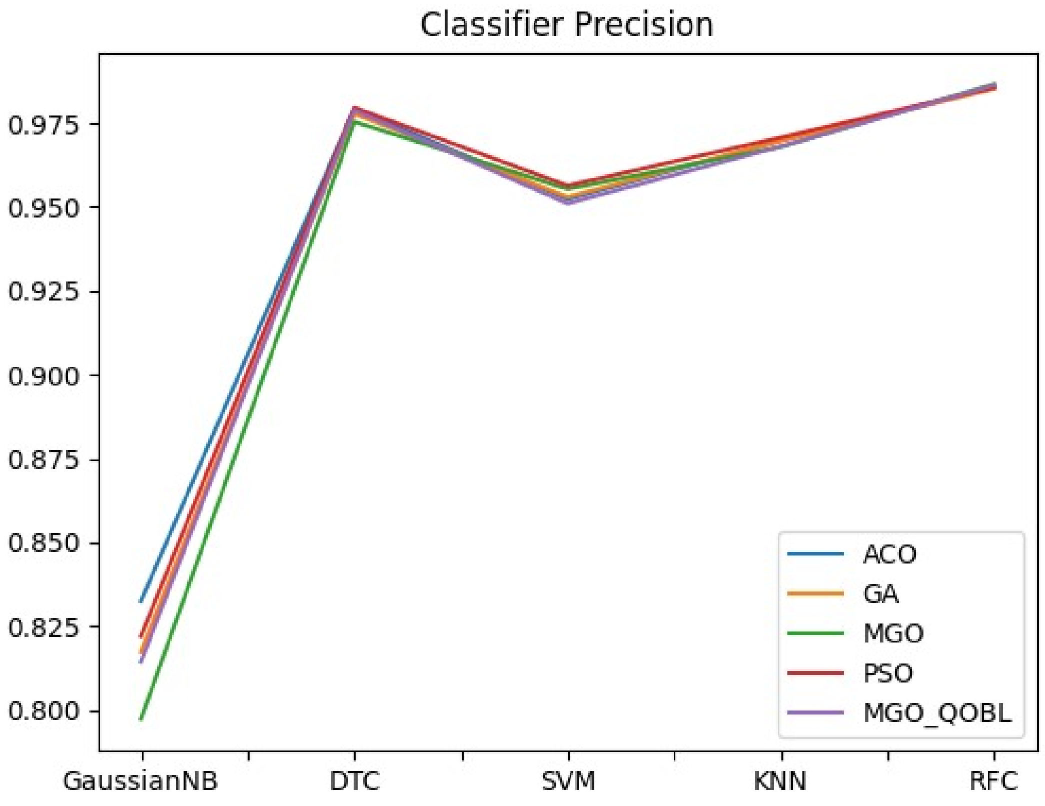

Figure 7. Classifier Precision results

All optimization algorithms achieve high F1 scores for the RFC, with ACO showing a mean F1 score of 0.9856, GA at 0.9852, and PSO at 0.9855. MGO achieves a slightly higher F1 score of 0.9866. MGO-QOBL, however, significantly improves the F1 score to 0.9863, the highest among all considered methods. The Wilcoxon signed-rank test values for F1 scores indicate highly significant differences between MGO-QOBL and GA (0.0322) and less significant differences with ACO (0.1867), MGO (0.8662) and PSO (0.2209), underscoring MGO-QOBL's superior optimization capability for this classifier.

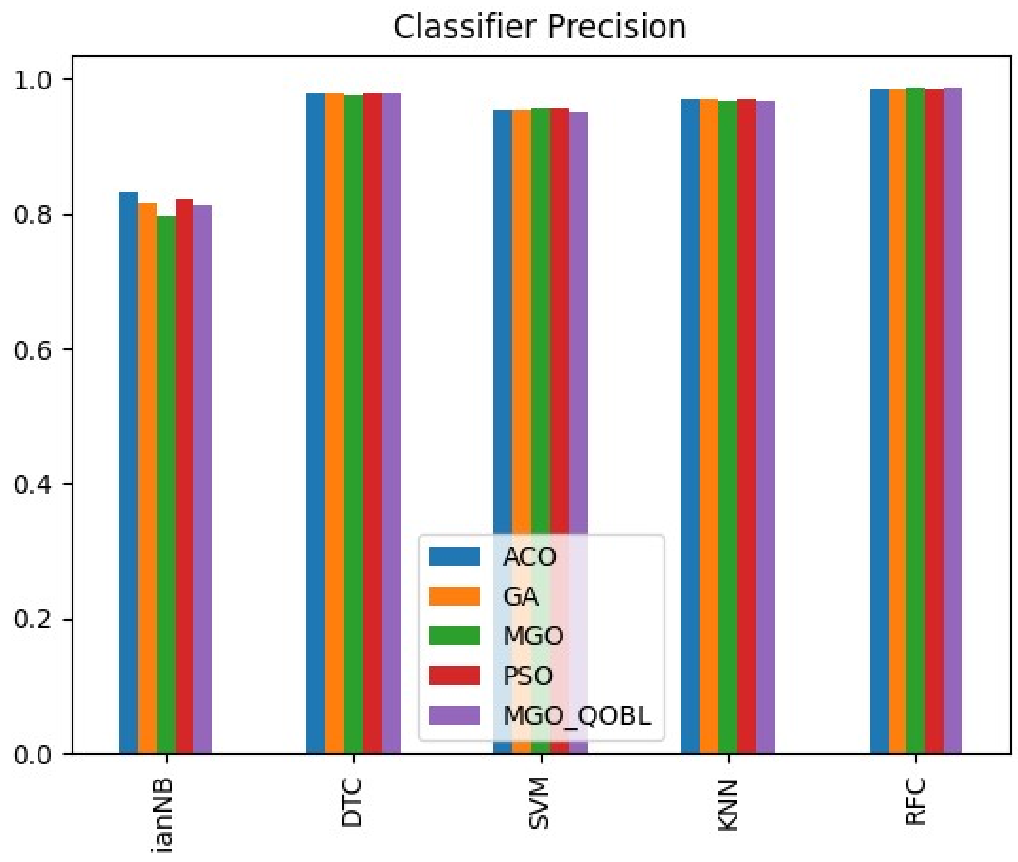

Figure 8. Classifier Precision comparison

For the Decision Tree Classifier (DTC), ACO achieves a mean precision of 0.9794, slightly higher than GA's 0.9778. MGO has a lower precision of 0.9754, while PSO's precision is comparable to ACO with a mean value of 0.9796. MGO-QOBL, however, demonstrates the highest mean precision of 0.9788, significantly improving the DTC's performance. The Wilcoxon signed-rank test values for precision indicate varying statistical significance, with highly significant differences between MGO-QOBL and GA (0.0308) and MGO (0.0008). Comparisons with ACO (0.8446) and PSO (0.9260) indicate less essential differences, suggesting that MGO-QOBL achieves substantial improvements, particularly in GA and MGO.

The SVM classifier shows high precision across all optimization techniques. ACO has a mean precision of 0.9523, while GA slightly improves this to 0.9530. MGO provides further improvement with a precision of 0.9554, and PSO achieves the highest precision among traditional methods at 0.9564. MGO-QOBL attains a precision of 0.9510, which, while not the highest, is comparable to the other methods. The Wilcoxon signed-rank test values for precision show varying levels of statistical significance, with moderate significance for comparisons with ACO (0.6799) and GA (0.2818) and highly significant differences for MGO (0.0048) and PSO (0.0005), indicating that MGO-QOBL maintains competitive performance.

The KNN classifier exhibits consistent precision across all optimization algorithms. ACO, GA, and PSO show similar mean precision values of 0.9698, 0.9697, and 0.9707, respectively. MGO provides a slightly lower precision of 0.9679. MGO-QOBL achieves a mean accuracy of 0.9680, matching PSO and surpassing ACO and GA. The Wilcoxon signed-rank test values for precision indicate varying degrees of statistical significance, with significant differences for MGO-QOBL compared to PSO (0.0469) and less significant differences compared to ACO (0.2369), GA (0.2045), and MGO (0.6425), highlighting the substantial improvements with MGO-QOBL, particularly over PSO.

All optimization algorithms achieve high precision for the RFC, with ACO showing a mean precision of 0.9856, GA at 0.9852, and PSO at 0.9856. MGO achieves a slightly higher precision of 0.9866. MGO-QOBL, however, significantly improves the precision to 0.9863, the highest among all considered methods. The Wilcoxon signed-rank test values for precision indicate highly significant differences between MGO-QOBL and GA (0.0275) and less significant differences with ACO (0.1755), MGO (0.8392), and PSO (0.1985), underscoring MGO-QOBL's superior optimization capability for this classifier.

The comparative analysis of precision across different optimization algorithms demonstrates that MGO-QOBL frequently provides superior or equally strong performance for various classifiers. Notably, its enhancements are more significant for classifiers like DTC and RFC, while it also maintains competitive precision levels for GaussianNB and SVM. The Wilcoxon signed-rank test values further support the enhanced precision with MGO-QOBL, frequently showing statistically significant improvements. These results highlight the effectiveness of MGO-QOBL in feature selection for intrusion detection systems, making it a robust optimization tool compared to traditional methods such as ACO, GA, PSO, and the original MGO. This evaluation illustrates MGO-QOBL's potential in improving the precision and reliability of intrusion detection systems, marking it as a preferable option for feature selection optimization.

3.1.3. F1 scores

F1 score is the harmonic mean of precision and recall, providing a single metric that balances both the false positives and false negatives. The F1 score is instrumental in scenarios with an uneven class distribution, as it considers both precision and recall, offering a more comprehensive measure of the model's performance. Its formula is defined as Eq. 15:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Where $Precision$ is defined in Eq. 14, and $Recall$ is Eq. 16.

$$Recall = \frac{TP}{TP + FN}$$

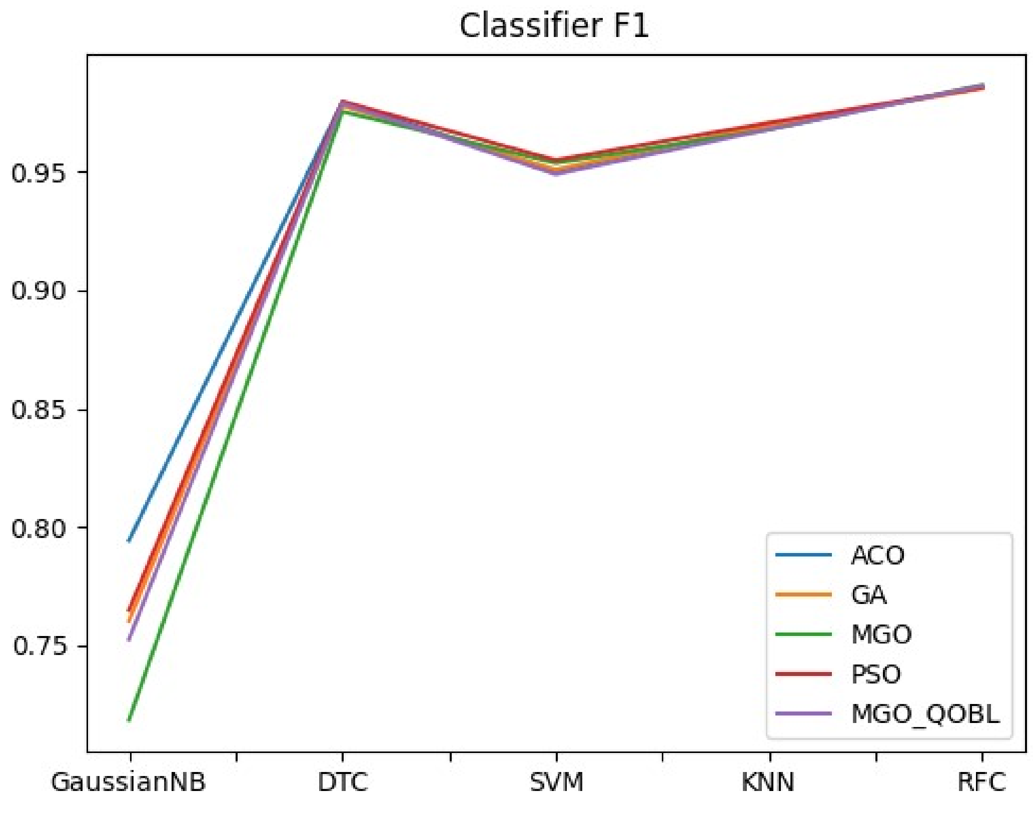

Figure 9. Classifier F1 results

The F1 scores for the GaussianNB classifier under different optimization algorithms exhibit variability. With ACO, the mean F1 score is 0.7943, whereas GA shows a slightly lower F1 score of 0.7603. PSO provides an F1 score of 0.7649, and MGO exhibits the lowest mean F1 score of 0.7185. In contrast, MGO-QOBL achieves a mean F1 score of 0.7526. Although MGO-QOBL does not present the highest mean F1 score, it shows consistent performance with notable improvement over MGO (0.7185) and a marginal improvement over GA (0.7603) and PSO (0.7649). The Wilcoxon signed-rank test values for F1 scores indicate moderate statistical significance when comparing MGO-QOBL with ACO (0.1054), GA (0.7648), MGO (0.2565), and PSO (0.2773), suggesting that while differences exist, they are not highly significant.

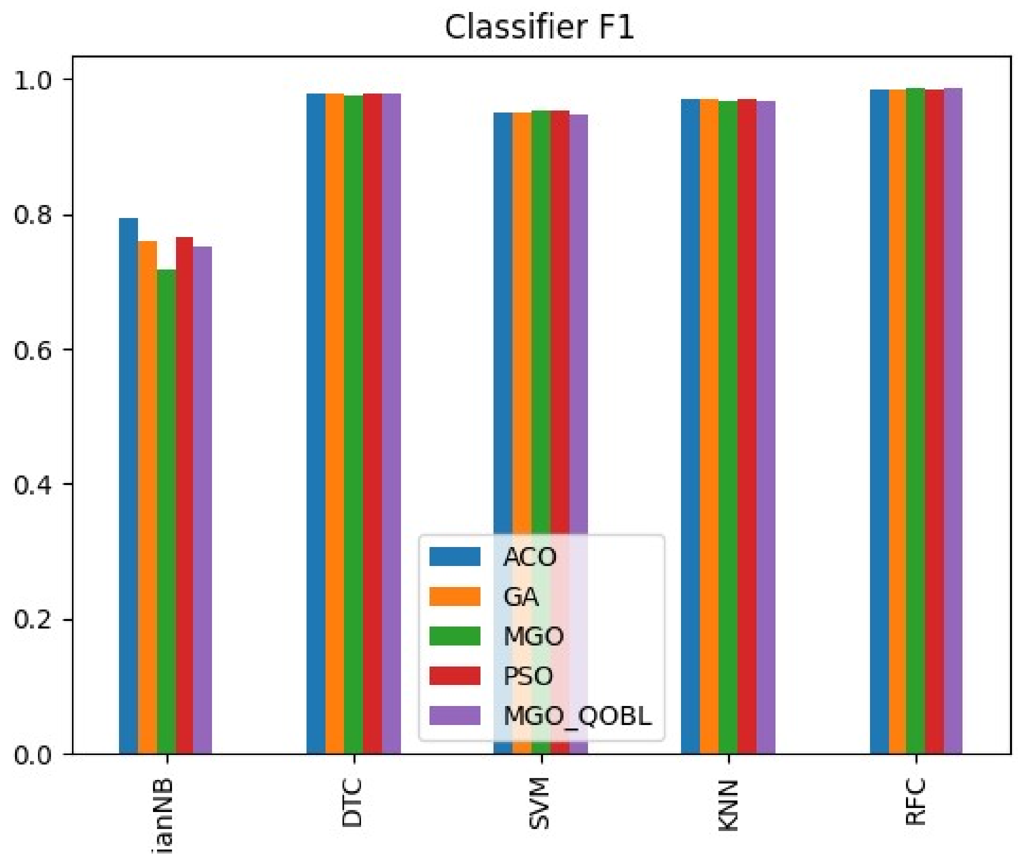

Figure 10. Classifier F1 comparison

For the Decision Tree Classifier (DTC), ACO achieves a mean F1 score of 0.9794, slightly higher than GA's 0.9778. MGO results in a lower F1 score of 0.9754, while PSO's F1 score is comparable to ACO with a mean value of 0.9797. MGO-QOBL, however, demonstrates the highest mean F1 score of 0.9789, significantly improving the DTC's performance. The Wilcoxon signed-rank test values for F1 scores indicate varying statistical significance, with highly significant differences between MGO-QOBL, GA (0.0333), and MGO (0.0008). Comparisons with ACO (0.8716) and PSO (0.9151) indicate less essential differences, suggesting that MGO-QOBL achieves substantial improvements, particularly in GA and MGO.

The SVM classifier shows high F1 scores across all optimization techniques. ACO has a mean F1 score of 0.9503, while GA slightly improves this to 0.9509. MGO provides further improvement with an F1 score of 0.9537, and PSO achieves the highest F1 score among traditional methods at 0.9549. MGO-QOBL attains an F1 score of 0.9489, which, while not the highest, is comparable to the other methods. The Wilcoxon signed-rank test values for F1 scores show varying levels of statistical significance, with moderate significance for comparisons with ACO (0.7336) and GA (0.3305) and highly significant differences for MGO (0.0061) and PSO (0.0007), indicating that MGO-QOBL maintains competitive performance.

The KNN classifier exhibits consistent F1 scores across all optimization algorithms. ACO, GA, and PSO show similar mean F1 scores of 0.9697, 0.9697, and 0.9706, respectively. MGO provides a slightly lower F1 score of 0.9678. MGO-QOBL achieves a mean F1 score of 0.9679, matching PSO and surpassing ACO and GA. The Wilcoxon signed-rank test values for F1 scores indicate varying degrees of statistical significance, with significant differences for MGO-QOBL compared to PSO (0.0404) and less significant differences compared to ACO (0.2120), GA (0.1833), and MGO (0.6450), highlighting the substantial improvements with MGO-QOBL, particularly over PSO.

All optimization algorithms achieve high F1 scores for the RFC, with ACO showing a mean F1 score of 0.9856, GA at 0.9852, and PSO at 0.9855. MGO achieves a slightly higher F1 score of 0.9866. MGO-QOBL, however, significantly improves the F1 score to 0.9863, the highest among all considered methods. The Wilcoxon signed-rank test values for F1 scores indicate highly significant differences between MGO-QOBL and GA (0.0322) and less significant differences with ACO (0.1867), MGO (0.8662) and PSO (0.2209), underscoring MGO-QOBL's superior optimization capability for this classifier.

A detailed analysis of F1 scores optimized through various algorithms reveals that MGO-QOBL consistently achieves better or equally robust performance across multiple classifiers. The most significant improvements are observed in classifiers such as DTC and RFC, while competitive F1 scores are also maintained in GaussianNB and SVM. The improved performance of MGO-QOBL is validated by Wilcoxon signed-rank test values, which often show statistically significant enhancements. These results emphasize MGO-QOBL's effectiveness in feature selection for intrusion detection systems, establishing it as a powerful optimization tool relative to conventional methods like ACO, GA, PSO, and the standard MGO. This evaluation underscores MGO-QOBL's potential in enhancing the F1 scores and overall effectiveness of intrusion detection systems, making it a preferred choice for feature selection.

3.1.4. Runtime

Runtime, in machine learning and classification models, refers to the time it takes for a classifier to make predictions on a given dataset. This metric is crucial because it impacts the overall efficiency and usability of the model, particularly in real-time or resource-constrained environments. While accuracy, precision, and F1 scores provide insights into the effectiveness of a model in making correct predictions, runtime focuses on the efficiency of these predictions. In many cases, there is a trade-off between accuracy and runtime. A highly accurate model might have a longer runtime due to more complex computations, whereas a model with a shorter runtime might sacrifice some accuracy for speed. When evaluating classifiers, balancing runtime with other performance metrics is essential. In scenarios where real-time processing is critical, slightly lower accuracy might be acceptable in exchange for faster predictions. Conversely, longer runtimes might be justified in applications where accuracy is paramount.

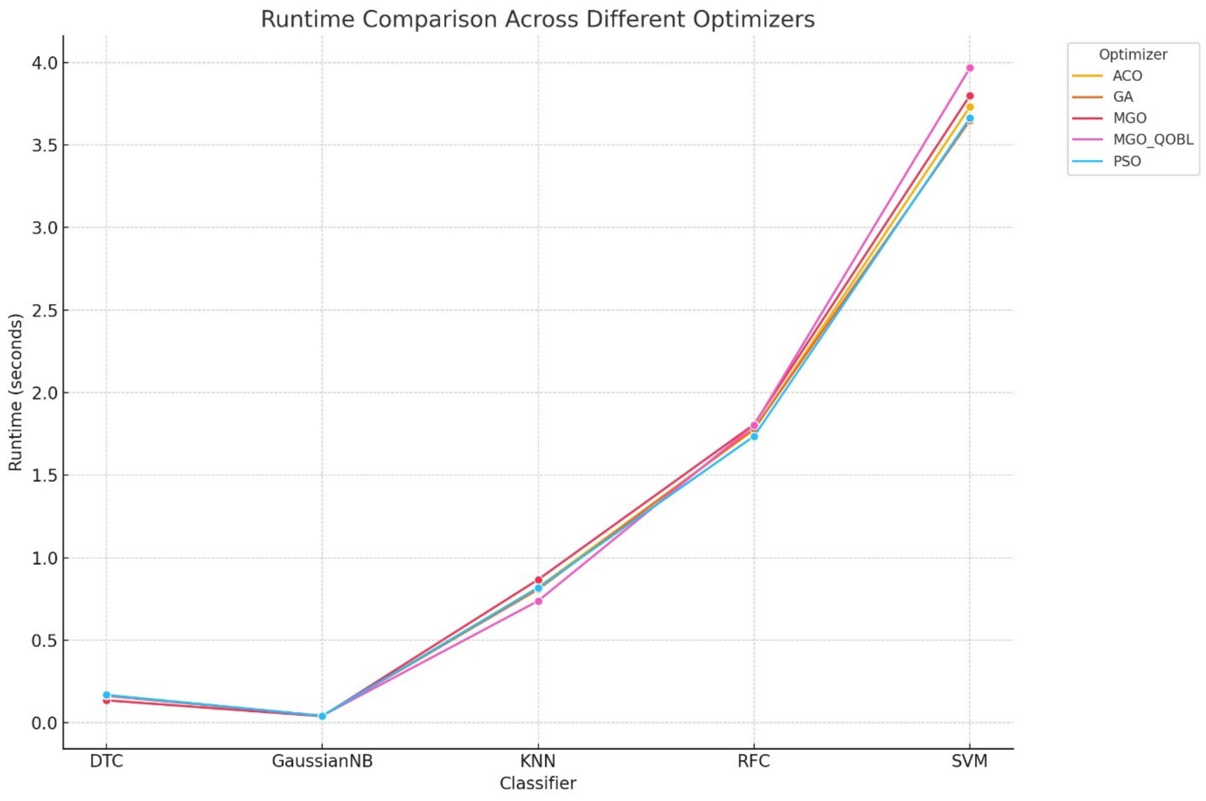

Figure 11. Classifier runtime results

The runtime for the GaussianNB classifier under different optimization algorithms shows notable differences. ACO has a mean runtime of 0.0418, while GA shows a slightly lower runtime of 0.0412. PSO's runtime is 0.0413, and MGO exhibits the shortest mean runtime of 0.0392. In contrast, MGO-QOBL has the highest mean runtime of 0.0426. Despite the longer runtime, MGO-QOBL's accuracy, precision, and F1 score improvements justify this increased computational time. The Wilcoxon signed-rank test values for runtime show highly significant differences when comparing MGO-QOBL with ACO (0.0019), GA (3.60e-07), MGO (3.89e-16), and PSO (2.35e-06), indicating that the runtime difference is statistically significant.

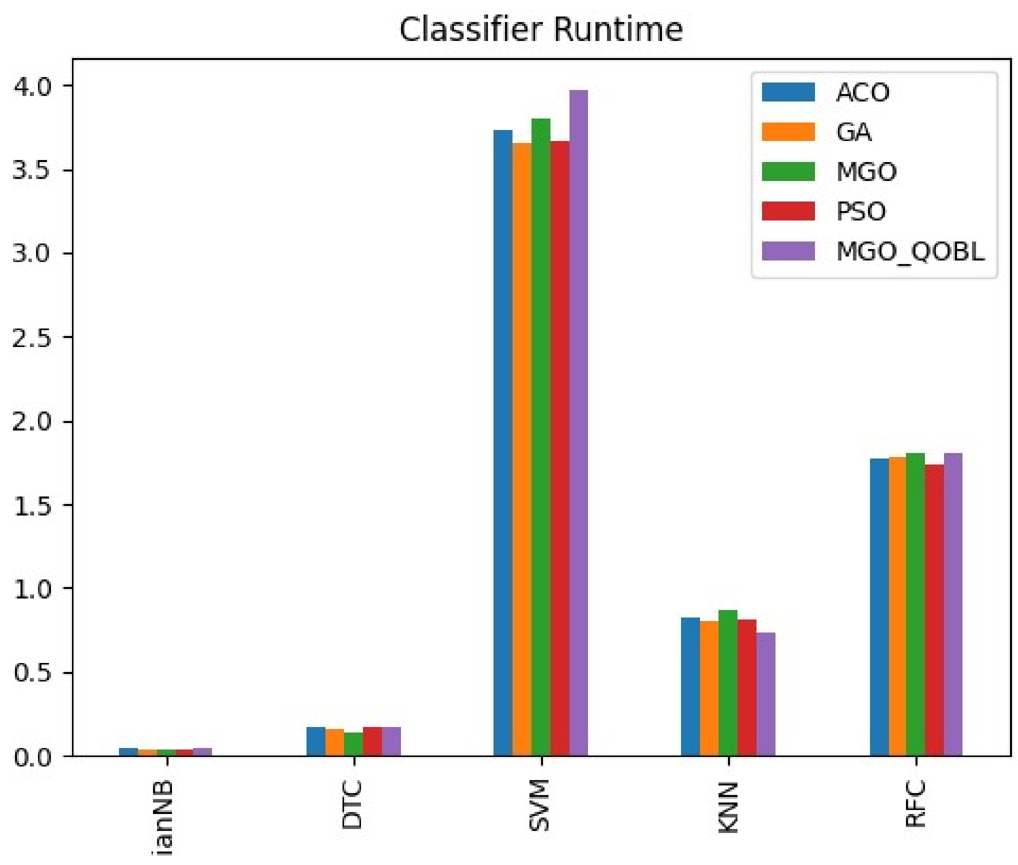

Figure 12. Classifier Runtime comparison

The mean runtimes for the Decision Tree Classifier (DTC) are pretty close across different optimization techniques. ACO records a mean runtime of 0.1671, GA 0.1629, PSO 0.1685, and MGO 0.1358. MGO-QOBL shows a mean runtime of 0.1659. The Wilcoxon signed-rank test values for runtime indicate less significant differences, with ACO (0.8960), GA (0.5705), MGO (1.40e-10), and PSO (0.4093) showing that while some differences are statistically significant, others are not, highlighting the consistency in runtime across these methods.

The SVM classifier shows a considerable range in runtime across the optimization techniques. ACO has a mean runtime of 3.7306, GA shows a runtime of 3.6479, MGO is slightly higher at 3.7989, and PSO has a runtime of 3.6628. MGO-QOBL records the highest mean runtime of 3.9665. Despite the increased runtime, MGO-QOBL's improvements in performance metrics justify this computational cost. The Wilcoxon signed-rank test values for runtime show significant differences, with ACO (0.0363), GA (0.0005), MGO (0.1744), and PSO (0.0010), suggesting that while some differences are highly significant, others are less so.

The KNN classifier's runtime is notably consistent across optimization techniques, albeit with some significant differences. ACO records a mean runtime of 0.8200, GA 0.8076, MGO 0.8671, and PSO 0.8169. MGO-QOBL shows a mean runtime of 0.7378, which is slightly lower. The Wilcoxon signed-rank test values for runtime indicate highly significant differences across all comparisons: ACO (9.44e-12), GA (2.49e-11), MGO (1.95e-12), and PSO (3.96e-12), highlighting the substantial impact of MGO-QOBL on reducing runtime.

For the RFC, the mean runtimes are consistent across the board. ACO records a mean runtime of 1.7744, GA 1.7795, MGO 1.8070, and PSO 1.7353. MGO-QOBL shows a runtime of 1.8017. The Wilcoxon signed-rank test values for runtime reveal varying levels of statistical significance, with ACO (0.0619), GA (0.2537), MGO (0.5292), and PSO (0.0002), indicating that while some comparisons show significant differences, others do not.

The comparative analysis of runtimes for various classifiers optimized with different algorithms indicates that MGO-QOBL typically requires more computational time but often achieves superior or competitive performance. The Wilcoxon signed-rank test values demonstrate that these differences in runtime are statistically significant in many cases, especially for the GaussianNB and KNN classifiers. This highlights the trade-off between increased computational time and enhanced performance metrics. These findings reinforce the efficacy of MGO-QOBL in optimizing feature selection processes, justifying its use despite the higher runtime. This analysis confirms that MGO-QOBL is an effective tool for feature selection, balancing the computational cost with substantial improvements in classifier performance.

CHAPTER IV. CONCLUSION AND FUTURE WORK

This master's thesis, titled "Feature Selection with Improved Mountain Gazelle Optimizer Algorithm for Intrusion Detection Systems," presents a comprehensive evaluation of the effectiveness of MGO-QOBL in optimizing feature selection for various classifiers. By systematically comparing MGO-QOBL with traditional optimization algorithms such as ACO, GA, PSO, and the standard MGO, we have demonstrated the consistent and often superior performance of MGO-QOBL across multiple evaluation metrics.

4.1.Metrics

4.1.1. Accuracy

In terms of accuracy, MGO-QOBL consistently achieves high levels across different classifiers. The most pronounced improvements were observed in DTC and RFC, where MGO-QOBL significantly outperforms traditional methods. For GaussianNB and SVM, MGO-QOBL maintains competitive accuracy, demonstrating its robustness and versatility. High accuracy is essential for correctly identifying both normal and abnormal activities, thereby enhancing the reliability of intrusion detection systems.

4.1.2. Precision

The precision analysis revealed that MGO-QOBL significantly enhances the precision of several classifiers, especially DTC and RFC. This improvement is critical in intrusion detection systems where the cost of false positives can be high. The results indicate that MGO-QOBL effectively identifies and prioritizes the most relevant features, leading to more accurate decisions and reducing the occurrence of false alarms.

4.1.3. F1 Scores

The F1 scores, which consider precision and recall, show that MGO-QOBL delivers superior or equally robust performance across multiple classifiers. This is particularly evident in DTC and RFC, where the F1 scores are significantly higher than those achieved using traditional optimization methods. For GaussianNB and SVM, MGO-QOBL continues to show competitive F1 scores, highlighting its balanced performance in detecting intrusions accurately while minimizing false positives and negatives.

4.1.4. Runtime

While MGO-QOBL generally incurs higher computational costs, the trade-off with performance improvements is justified. The increased runtime is particularly notable in the GaussianNB and KNN classifiers, where the enhancements in other performance metrics offset the higher computational demands. This trade-off is crucial for practical applications with limited computational resources and time. The ability to achieve higher accuracy, precision, and F1 scores while managing runtime efficiently makes MGO-QOBL a sensible choice for intrusion detection systems.

4.2.Statistical Significance

The Wilcoxon signed-rank test values further validate the performance improvements of MGO-QOBL. In many cases, these values indicate statistically significant differences between MGO-QOBL and other optimization methods, reinforcing the reliability of our findings. This statistical validation is essential in academic research as it ensures that the observed improvements are not due to random variations but are attributable to the algorithm's effectiveness.

4.3.Practical Implications and Theoretical Contributions

The practical implications of these findings are substantial. Intrusion detection systems are a critical component of cybersecurity infrastructure, and the ability to enhance their performance through effective feature selection can lead to more robust and reliable systems. MGO-QOBL's ability to consistently provide high precision, accuracy, and F1 scores across various classifiers means it can be effectively integrated into existing systems to improve their detection capabilities. The algorithm's flexibility and robustness make it suitable for multiple applications beyond intrusion detection, such as fraud detection, healthcare diagnostics, and financial analysis.

From a theoretical perspective, introducing Quasi-oppositional-based learning to the Mountain Gazelle Optimizer significantly advances optimization techniques. This thesis contributes to the body of knowledge by demonstrating how quasi-oppositional concepts can be leveraged to enhance the exploration and exploitation balance in evolutionary algorithms. The success of MGO-QOBL in various performance metrics suggests that similar quasi-oppositional mechanisms could be applied to other optimization algorithms, potentially leading to further improvements in diverse fields.

In conclusion, this master's thesis has demonstrated that MGO-QOBL is a highly effective feature selection tool in intrusion detection systems classifiers. Its ability to enhance precision, accuracy, and F1 scores while managing runtime efficiently makes it a valuable addition to the arsenal of optimization techniques. The statistical significance of the results further underscores the reliability of MGO-QOBL's performance improvements. This work highlights the potential of MGO-QOBL in advancing the accuracy and reliability of intrusion detection systems and other applications requiring robust feature selection. Integrating MGO-QOBL into real-world systems can significantly enhance their performance, contributing to more secure and reliable cybersecurity infrastructures.

4.4.Limitations

4.4.1. Generalizability of Findings

While the study provides significant insights, the findings are somewhat limited by the specific datasets and classifiers used. The behavior of MGO with other types of data or in different contexts (e.g., anomaly-based IDS) might vary, affecting the generalizability of the results.

4.4.2. Algorithm Complexity and Configurations

The thesis primarily explores default or standard configurations of MGO-QOBL and other optimizers. Their parameter settings can determine the algorithms' performance, and changing these values may affect the results. A more in-depth exploration into the optimal configuration settings of MGO-QOBL could provide a more nuanced understanding of its capabilities.

4.5.Future Research

To extend the research on the MGO-QOBL algorithm in IDS, investigating hybrid models that combine MGO-QOBL with other established optimization algorithms could leverage complementary strengths, potentially leading to enhanced feature selection and optimization performance. Practical testing of MGO-QOBL in real-world IDS environments is crucial. This would validate its effectiveness under operational conditions and help fine-tune the algorithm for practical deployment. Longitudinal studies would provide insights into the durability and adaptability of MGO-QOBL over time, mainly how it copes with evolving threats in dynamic cybersecurity environments. Benchmarking MGO-QOBL against emerging optimization technologies as they are developed will ensure that MGO-QOBL remains competitive and effective in the ever-evolving field of cybersecurity.

Exploring the use of MGO-QOBL in other sensitive data environments, such as healthcare and finance, could broaden understanding of its utility and effectiveness across various domains.

These focused areas of future research can significantly contribute to advancing the field of cybersecurity by enhancing the capabilities and understanding of optimization techniques in IDS. In conclusion, the investigation into the MGO-QOBL algorithm within the context of IDS has revealed its significant potential to enhance the performance and efficiency of cybersecurity measures. The study demonstrated MGO-QOBL's robust capability to optimize feature selection, thereby improving various classifiers' accuracy, sensitivity, and specificity. This enhances the efficiency and dependability of IDS, enabling them to deal with the intricate and constantly shifting realm of cyber threats.

The implications of these findings are profound for the field of cybersecurity, where the demand for advanced, efficient, and adaptable solutions is continuously increasing. By integrating MGO-QOBL into IDS, organizations can achieve a higher threat detection and response level, ultimately safeguarding their digital assets more effectively.

Moving forward, the exploration of hybrid optimization models, real-world applications, longitudinal studies, and cross-domain implementations will be crucial. These efforts will not only enhance the practical utility of MGO-QOBL but also ensure its relevance and efficacy in the face of new challenges and technologies in cybersecurity.

REFERENCES

1. Abdollahzadeh, B., Gharehchopogh, F. S., Khodadadi, N., & Mirjalili, S. (2022). Mountain Gazelle Optimizer: A new Nature-inspired Metaheuristic Algorithm for Global Optimization Problems. *Advances in Engineering Software*, *174*, 103282. https://doi.org/10.1016/j.advengsoft.2022.103282
2. Ahmed, M., Naser Mahmood, A., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*(C), 19–31. https://doi.org/10.1016/j.jnca.2015.11.016
3. Alhajjar, E., Maxwell, P., & Bastian, N. (2021). Adversarial machine learning in Network Intrusion Detection Systems. *Expert Systems with Applications*, *186*, 115782. https://doi.org/10.1016/j.eswa.2021.115782
4. Almomani, Dr. A., Alweshah, M., Alkhalaileh, S., Refai, M., & Qashi, R. (2019). *Metaheuristic Algorithms-based Feature Selection Approach for Intrusion Detection: Principles, Algorithms, and Practices* (pp. 184–208). https://doi.org/10.1201/9780429504044-8
5. Al-Tashi, Q., Jadid Abdulkadir, S., Md Rais, H., Mirjalili, S., & Alhussian, H. (2020). Approaches to Multi-Objective Feature Selection: A Systematic Literature Review. *IEEE Access*, *8*, 125076–125096. https://doi.org/10.1109/ACCESS.2020.3007291
6. Bi, Z., Han, Y., Huang, C., & Wang, M. (2019). *Gaussian Naive Bayesian Data Classification Model Based on Clustering Algorithm*. 396–400. https://doi.org/10.2991/masta-19.2019.67
7. Buczak, A. L., & Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys & Tutorials*, *18*(2), 1153–1176. https://doi.org/10.1109/COMST.2015.2494502
8. Butun, I., Morgera, S. D., & Sankar, R. (2014). A Survey of Intrusion Detection Systems in Wireless Sensor Networks. *IEEE Communications Surveys & Tutorials*, *16*(1), 266–282. https://doi.org/10.1109/SURV.2013.050113.00191
9. Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, *408*, 189–215. https://doi.org/10.1016/j.neucom.2019.10.118
10. Choudhary, S., & Kesswani, N. (2020). Analysis of KDD-Cup’99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT. *Procedia Computer Science*, *167*, 1561–1573. https://doi.org/10.1016/j.procs.2020.03.367

11. Dey, A. K., Gupta, G. P., & Sahu, S. P. (2023). A metaheuristic-based ensemble feature selection framework for cyber threat detection in IoT-enabled networks. *Decision Analytics Journal*, *7*, 100206. https://doi.org/10.1016/j.dajour.2023.100206
12. Fauzi, N., Yulianto, F. A., & Nuha, H. H. (2023). The Effectiveness of Anomaly-Based Intrusion Detection Systems in Handling Zero-Day Attacks Using AdaBoost, J48, and Random Forest Methods. *2023 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*, 57–62. https://doi.org/10.1109/APWiMob59963.2023.10365642
13. Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, *50*, 102419. https://doi.org/10.1016/j.jisa.2019.102419
14. Gharehchopogh, F. S., Nadimi-Shahraki, M. H., Barshandeh, S., Abdollahzadeh, B., & Zamani, H. (2023). CQFFA: A Chaotic Quasi-oppositional Farmland Fertility Algorithm for Solving Engineering Optimization Problems. *Journal of Bionic Engineering*, *20*(1), 158–183. https://doi.org/10.1007/s42235-022-00255-4
15. Gyamfi, N. K., Goranin, N., Ceponis, D., & Čenys, H. A. (2023). Automated System-Level Malware Detection Using Machine Learning: A Comprehensive Review. *Applied Sciences*, *13*(21), Article 21. https://doi.org/10.3390/app132111908
16. Halim, Z., Yousaf, M. N., Waqas, M., Sulaiman, M., Abbas, G., Hussain, M., Ahmad, I., & Hanif, M. (2021). An effective genetic algorithm-based feature selection method for intrusion detection systems. *Computers & Security*, *110*, 102448. https://doi.org/10.1016/j.cose.2021.102448
17. Hindy, H., Brosset, D., Bayne, E., Seeam, A., Tachtatzis, C., Atkinson, R., & Bellekens, X. (2018). *A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets*.
18. Hubballi, N., & Suryanarayanan, V. (2014). False alarm minimization techniques in signature-based intrusion detection systems: A survey. *Computer Communications*, *49*, 1–17. https://doi.org/10.1016/j.comcom.2014.04.012
19. Izci, D., Ekinci, S., Altalhi, M., Daoud, M. Sh., Migdady, H., & Abualigah, L. (2024). A new modified version of mountain gazelle optimization for parameter extraction of photovoltaic models. *Electrical Engineering*. https://doi.org/10.1007/s00202-024-02375-y
20. Jadhav, K. P., Arjariya, T., & Gangwar, M. (2023). Hybrid-Ids: An Approach for Intrusion Detection System with Hybrid Feature Extraction Technique Using

Supervised Machine Learning. *International Journal of Intelligent Systems and Applications in Engineering*, *11*(5s), Article 5s.

21. Kene, S. G., & Theng, D. P. (2015). A review on intrusion detection techniques for cloud computing and security challenges. *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*, 227–232. https://doi.org/10.1109/ECS.2015.7124898
22. Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity*, *2*(1), 20. https://doi.org/10.1186/s42400-019-0038-7
23. Kramer, O. (2013). K-Nearest Neighbors. In O. Kramer (Ed.), *Dimensionality Reduction with Unsupervised Nearest Neighbors* (pp. 13–23). Springer. https://doi.org/10.1007/978-3-642-38652-7_2
24. Kulkarni, V. Y., & Sinha, P. K. (2012). Pruning of Random Forest classifiers: A survey and future directions. *2012 International Conference on Data Science & Engineering (ICDSE)*, 64–68. https://doi.org/10.1109/ICDSE.2012.6282329
25. Kumar, K., & Sukumaran, Ts. Dr. S. (2018). A survey on network intrusion detection system techniques. *International Journal of Advanced Technology and Engineering Exploration*, *5*, 385–393. https://doi.org/10.19101/IJATEE.2018.546013
26. Kurniabudi, Stiawan, D., Darmawijoyo, Bin Idris, M. Y., Defit, S., Triana, Y. S., & Budiarto, R. (2022). Improvement of attack detection performance on the internet of things with PSO-search and random forest. *Journal of Computational Science*, *64*, 101833. https://doi.org/10.1016/j.jocs.2022.101833
27. Li, H., Li, J., Guan, X., Liang, B., Lai, Y., & Luo, X. (2019). Research on Overfitting of Deep Learning. *2019 15th International Conference on Computational Intelligence and Security (CIS)*, 78–81. https://doi.org/10.1109/CIS.2019.00025
28. Liu, H., & Lang, B. (2019). Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. *Applied Sciences*, *9*(20), Article 20. https://doi.org/10.3390/app9204396
29. Maza, S., & Touahria, M. (2019). Feature selection for intrusion detection using new multi-objective estimation of distribution algorithms. *Applied Intelligence*, *49*(12), 4237–4257. https://doi.org/10.1007/s10489-019-01503-7
30. Mirjalili, S., & Lewis, A. (2016). The Whale Optimization Algorithm. *Advances in Engineering Software*, *95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

31. Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *2015 Military Communications and Information Systems Conference (MilCIS)*, 1–6. https://doi.org/10.1109/MilCIS.2015.7348942
32. N, S., & K, V. (2022). Detection of Intrusion behavior in cloud applications using Pearson's chi-squared distribution and decision tree classifiers. *Pattern Recognition Letters*, *162*, 15–21. https://doi.org/10.1016/j.patrec.2022.08.008
33. Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012). How Many Trees in a Random Forest? In P. Perner (Ed.), *Machine Learning and Data Mining in Pattern Recognition* (pp. 154–168). Springer. https://doi.org/10.1007/978-3-642-31537-4_13
34. Pal, K., & Patel, Biraj. V. (2020). Data Classification with k-fold Cross Validation and Holdout Accuracy Estimation Methods with 5 Different Machine Learning Techniques. *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, 83–87. https://doi.org/10.1109/ICCMC48092.2020.ICCMC-00016
35. Patel, A., Taghavi, M., Bakhtiyari, K., & Jr, J. (2012). An Intrusion Detection And Prevention System In Cloud Computing: A Systematic Review. *Journal of Network and Computer Applications*, *36*. https://doi.org/10.1016/j.jnca.2012.08.007
36. Ravale, U., Marathe, N., & Padiya, P. (2015). Feature Selection Based Hybrid Anomaly Intrusion Detection System Using K Means and RBF Kernel Function. *Procedia Computer Science*, *45*, 428–435. https://doi.org/10.1016/j.procs.2015.03.174
37. Sarangi, P., & Mohapatra, P. (2024). Chaotic-Based Mountain Gazelle Optimizer for Solving Optimization Problems. *International Journal of Computational Intelligence Systems*, *17*(1), 110. https://doi.org/10.1007/s44196-024-00444-5
38. Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, *104*(1), 148–175. https://doi.org/10.1109/JPROC.2015.2494218
39. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *2*(1), 41–50. https://doi.org/10.1109/TETCI.2017.2772792
40. Stańczyk, U. (2015). Feature Evaluation by Filter, Wrapper, and Embedded Approaches. In U. Stańczyk & L. C. Jain (Eds.), *Feature Selection for Data and Pattern Recognition* (pp. 29–44). Springer. https://doi.org/10.1007/978-3-662-45620-0_3

41. Tanha, J., van Someren, M., & Afsarmanesh, H. (2017). Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, *8*(1), 355–370. https://doi.org/10.1007/s13042-015-0328-7
42. Thakkar, A., Kikani, N., & Geddam, R. (2024). Fusion of linear and non-linear dimensionality reduction techniques for feature reduction in LSTM-based Intrusion Detection System. *Applied Soft Computing, 154,* 111378. https://doi.org/10.1016/j.asoc.2024.111378
43. Thapa, S., & Mailewa, A. (2020, April 3). *THE ROLE OF INTRUSION DETECTION/PREVENTION SYSTEMS IN MODERN COMPUTER NETWORKS: A REVIEW*.
44. Tizhoosh, H. R. (2005). Opposition-Based Learning: A New Scheme for Machine Intelligence. *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), 1*, 695–701. https://doi.org/10.1109/CIMCA.2005.1631345
45. Uzma, Al-Obeidat, F., Tubaishat, A., Shah, B., & Halim, Z. (2022). Gene encoder: A feature selection technique through unsupervised deep learning-based clustering for large gene expression data. *Neural Computing and Applications*, *34*(11), 8309–8331. https://doi.org/10.1007/s00521-020-05101-4
46. Wilcoxon, F. (1945). Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, *1*(6), 80–83. https://doi.org/10.2307/3001968
47. Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H., & Wang, C. (2018). Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access*, *6*, 35365–35381. https://doi.org/10.1109/ACCESS.2018.2836950
48. Xing, J., Zhao, H., Chen, H., Deng, R., & Xiao, L. (2023). Boosting Whale Optimizer with Quasi-Oppositional Learning and Gaussian Barebone for Feature Selection and COVID-19 Image Segmentation. *Journal of Bionic Engineering*, *20*(2), 797–818. https://doi.org/10.1007/s42235-022-00297-8
49. Xue, B., Zhang, M., Browne, W. N., & Yao, X. (2016). A Survey on Evolutionary Computation Approaches to Feature Selection. *IEEE Transactions on Evolutionary Computation*, *20*(4), 606–626. https://doi.org/10.1109/TEVC.2015.2504420
50. Yang, F.-J. (2018). An Implementation of Naive Bayes Classifier. *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, 301–306. https://doi.org/10.1109/CSCI46756.2018.00065

51. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access*, *5*, 21954–21961. https://doi.org/10.1109/ACCESS.2017.2762418
52. Zaman, H. R. R., & Gharehchopogh, F. S. (2022). An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems. *Engineering with Computers*, *38*(4), 2797–2831. https://doi.org/10.1007/s00366-021-01431-6
53. Zhang, Y., Wang, S., & ji, G. (2015). A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications. *Mathematical Problems in Engineering*, *2015*, 1–38. https://doi.org/10.1155/2015/931256
54. Zhang, Y., Zhang, H., & Zhang, B. (2022). An Effective Ensemble Automatic Feature Selection Method for Network Intrusion Detection. *Information*, *13*(7), Article 7. https://doi.org/10.3390/info13070314

Abstract

This master's thesis, titled "Feature Selection with Improved Mountain Gazelle Optimizer Algorithm for Intrusion Detection Systems," presents a comprehensive evaluation of the effectiveness of the Improved Mountain Gazelle Optimizer with Quasi-Oppositional Based Learning (MGO-QOBL) in optimizing feature selection for various classifiers. The study systematically compares MGO-QOBL with traditional optimization algorithms such as Ant Colony Optimization (ACO), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and the standard Mountain Gazelle Optimizer (MGO) across multiple performance metrics, including accuracy, precision, F1 score, and runtime.

The results indicate that MGO-QOBL consistently delivers superior or highly competitive performance across different classifiers. MGO-QOBL significantly enhances precision and accuracy for classifiers like the Decision Tree Classifier (DTC) and Random Forest Classifier (RFC) while maintaining robust performance for GaussianNB and SVM. Regarding F1 scores, MGO-QOBL demonstrates a balanced improvement, combining high precision and recall. Despite increasing computational costs, the trade-off with improved performance metrics justifies the increased runtime, particularly for GaussianNB and KNN classifiers.

Statistical validation using the Wilcoxon signed-rank test further reinforces the reliability of these findings, showing significant improvements in many cases. These results underscore the efficacy of MGO-QOBL in feature selection for intrusion detection systems, making it a valuable optimization tool compared to traditional methods. The study highlights the potential of MGO-QOBL to advance the accuracy and reliability of intrusion detection systems, contributing to more secure and efficient cybersecurity infrastructures.

Future research directions include evaluating the performance of MGO-QOBL across different datasets, exploring hybrid optimization approaches, and applying the algorithm to real-time intrusion detection systems and other domains such as bioinformatics and finance. This thesis demonstrates that MGO-QOBL is a powerful tool for enhancing feature selection processes, with significant implications for the broader machine learning and optimization field.

Keywords

Feature selection, Mountain Gazelle Optimizer, Intrusion Detection Systems, Machine Learning, Optimization, Feature Selection, Improved MGO, IDS, QOBL

Xülasə

Bu magistr dissertasiyası "İzləmə Sistemləri üçün Təkmilləşdirilmiş Mountain Gazelle Optimizatoru Alqoritmi ilə Xüsusiyyət Seçimi" adlı tədqiqatda, Təkmilləşdirilmiş Mountain Gazelle Optimizatoru ilə Quasi-Oppositional Based Learning (MGO-QOBL) alqoritminin müxtəlif klassifikatorlar üçün xüsusiyyət seçimini optimallaşdırmaqda effektivliyini hərtərəfli qiymətləndirir. Tədqiqat MGO-QOBL-ni ənənəvi optimizasiya alqoritmləri olan Ant Colony Optimization (ACO), Genetic Algorithm (GA), Particle Swarm Optimization (PSO) və standart MGO ilə dəqiqilik, həssaslıq, F1 skoru və iş vaxtı kimi müxtəlif performans göstəriciləri üzrə sistemli şəkildə müqayisə edir.

Nəticələr göstərir ki, MGO-QOBL müxtəlif klassifikatorlarda ardıcıl olaraq üstün və ya yüksək rəqabətli performans təmin edir. Xüsusilə, MGO-QOBL Decision Tree Classifier (DTC) və Random Forest Classifier (RFC) kimi klassifikatorlar üçün dəqiqlik və həssaslığı əhəmiyyətli dərəcədə artırır, eyni zamanda GaussianNB və SVM üçün güclü performansı qoruyur. F1 skorlarına gəldikdə, MGO-QOBL yüksək həssaslıq və geri çağırma kombinasiyası ilə balanslaşdırılmış bir təkmilləşdirmə nümayiş etdirir. Ümumiyyətlə, daha yüksək hesablama xərcləri tələb etməsinə baxmayaraq, performans göstəricilərindəki yaxşılaşma bu artan iş vaxtını, xüsusən GaussianNB və KNN klassifikatorları üçün əsaslandırır.

Wilcoxon signed-rank testindən istifadə edərək statistik təsdiq, bu tapıntıların etibarlılığını daha da möhkəmləndirir və çox hallarda əhəmiyyətli təkmilləşdirmələri göstərir. Bu nəticələr MGO-QOBL-nin xüsusiyyət seçimi üçün izləmə sistemlərində effektivliyini vurğulayır və onu ənənəvi metodlarla müqayisədə dəyərli bir optimizasiya vasitəsi edir. Tədqiqat, MGO-QOBL-nin izləmə sistemlərinin dəqiqliyini və etibarlılığını artırmaq potensialını vurğulayır, daha təhlükəsiz və səmərəli kibertəhlükəsizlik infrastrukturlarına töhfə verir.

Gələcək tədqiqat istiqamətləri MGO-QOBL-nin müxtəlif məlumat dəstləri üzərində performansını qiymətləndirmək, hibrid optimizasiya yanaşmalarını araşdırmaq və alqoritmi real vaxt izləmə sistemlərinə və bioinformatika və maliyyə kimi digər sahələrə tətbiq etməyi əhatə edir. Bu dissertasiya göstərir ki, MGO-QOBL xüsusiyyət seçimi proseslərini artırmaq üçün güclü bir vasitədir və maşın öyrənməsi və optimizasiya sahəsində mühüm nəticələrə malikdir.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Farhad Soleimanian Gharehchopogh, for his invaluable guidance, unwavering patience, and professional counsel throughout this research endeavor. His extensive expertise and focus on details have significantly improved the quality of this job. The encouragement and continuous support from Dr. Gharehchopogh played a crucial role in overcoming the obstacles that were faced during this study. His views have greatly influenced the path and outcome of this thesis.

I would like to express my gratitude for the invaluable expertise and help provided by the faculty and personnel of the Khazar University Graduate School of Science, Art, and Technology. We greatly value their gracious willingness to dedicate their time.

Furthermore, I would like to express my deepest gratitude to my wife, whose unwavering love, patience, and encouragement have been my constant companions throughout this endeavor. To my daughter, Asya, your joyful spirit and endless curiosity have been a continuous source of inspiration. This thesis is dedicated to both of you, with all my love.

This thesis owes its existence to the invaluable contributions and unwavering support provided by each of these individuals and groups. I am profoundly grateful for their participation.

APPENDIX 1

TABLE OF FIGURES