Journal of Theoretical and Applied Information Technology
31st August 2024. Vol.102. No. 16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

IMPROVING PORT SCAN CYBERSECURITY RISKS DETECTION USING FEATURES SELECTION TECHNIQUES WITH ML ALGORITHMS

RAMI SHEHAB[1], RANA ALRAWASHDEH [2], ROMEL AL-ALI [3], TAYSEER ALKHDOUR [4], MOHAMMED AMIN ALMAIAH[5]

[1]College of Computer Sciences and Information Technology, King Faisal University, Al-Ahsa 31982, Saudi Arabia

[2] King Fahd of Petroleum and Mineral, Faculty of computer science and information system, Dhahran 31261, Saudi Arabia

[3] Associate Professor, the National Research Center for Giftedness and Creativity, King Faisal University, Saudi Arabia

[4] College of Computer Sciences and Information Technology, King Faisal University, Al-Ahsa 31982, Saudi Arabia

[5] King Abdullah the II IT School, The University of Jordan, Amman 11942, Jordan

[*]Corresponding Author: Rtshehab@kfu.edu.sa and m.almaiah@ju.edu.jo

ABSTRACT

Malicious automated tools use port scan attacks to explore a target systems network ports aiming to find open ports and potential weaknesses. Port scanning can serve as a tool for system admins and cybersecurity experts to explore the main weaknesses in the network. Several countermeasures have been employed to defend against port attacks such as firewalls, intrusion detection systems (IDS) and network monitoring tools. These countermeasures aim to prevent malicious port scanning attacks. Based on that, port scan risks assessment is one of the essential step for detecting threats, vulnerabilities and protect the network security. Thus, this work aims to detect port scan attacks using features selection techniques with machine learning (ML) algorithms to reduce cyber-attacks being successful. To achieve this objective, we used three feature selection methods namely, ant colony algorithm (ACO), genetic algorithm (GA), and gray wolf optimization (GWO) with machine learning algorithms such as support vector machine (SVM), and nearest neighbor (KNN). The proposed work has been evaluated using confusion matrix measurements in terms of precision, recall, F1 score and accuracy. The study findings show that the percentage of risks and attacks detection over 99% for all proposed models. This study confirms that through the use of feature selection algorithms and machine learning methods, can help researchers to identify port behaviors and attacks in more efficiently.

Keywords: *Port Scan Attacks; Cyber-Risk; Machine Learning; Feature Selections; ACO; GA; GWO; SVM; KNN.*

1 INTRODUCTION

A port scan attack is a method used to investigate a system or network, for ports and services. Malicious actors send data packets to ports to check their status and collect details about the vulnerabilities of the targeted system [1]. Various forms of port attacks include TCP port scanning, UDP port scanning, covert port scanning and SYN flood attacks [5]. The aim of these attacks is to gather information for entry or future sophisticated assaults. Detecting and preventing port scan attacks is crucial for network security and tools like intrusion detection systems, firewalls and network monitoring solutions play a role, in spotting and stopping activities to protect network assets [4]. Detecting port scan attacks plays a role, in upholding network security by spotting threats at a

stage and enabling proactive actions to be taken [2]. It assists in evaluating weaknesses thwarting entry and reducing the impact of attacks. Early detection allows companies to enact response protocols swiftly ensuring adherence, to regulations and enhancing network efficiency. Moreover, it fosters a culture of security awareness supports learning from incidents and empowers organizations to safeguard their systems, data and assets efficiently [3]. Port scanning activities serve purposes whether for lawful intentions. They are employed for tasks such, as scouting, identifying weaknesses testing security systems conducting penetration tests, mapping networks and researching security measures. These scans aid attackers in collecting data, pinpointing vulnerabilities, exploiting flaws assessing security protocols mimicking attacks charting out network structures and delving into security practices [16]. Nonetheless it is essential to carry out port scans and, with authorization since unauthorized or malicious usage is both unlawful and unethical [17].

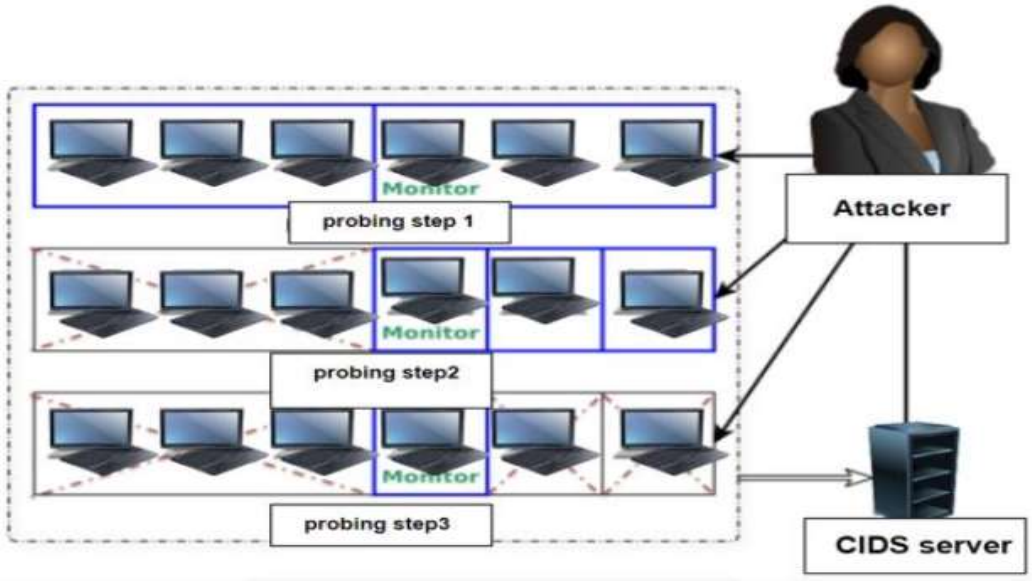

Figure 1: Probe Attack Umbrella.

The goals of identifying port scans, through feature selection and machine learning methods are to spot attacks correctly categorize network activity use resources effectively adjust to changing threats detect issues in time decrease false alarms and scale as needed [7]. By using feature selection algorithms and machine learning models companies can improve the efficiency and precision of port detection. This helps in responding and taking actions to prevent security breaches [6]. Detecting port scan attacks is essential. It has its challenges. Attackers employ tactics making detection difficult with encrypted traffic. There can be instances of positives or negatives [9]. The complexity of networks adds to the difficulty of distinguishing threats and staying updated on evolving attack methods is crucial. The intensive analysis can impact network performance and privacy issues need attention. Without details accuracy, in detection may [10]. Zero day attacks present a significant hurdle. System overload may result in delays or missed detections. Despite these challenges organizations can utilize techniques, like network monitoring and behavioral analysis to boost port detection and enhance security [12] [14].

Detecting port scan attacks is driven by the necessity to uphold network security and safeguard systems [13]. It aids in the detection of threats wards off entry and pinpoints vulnerabilities, for remediation. Other reasons include curtailing attacks adhering to regulations and enhancing network efficiency [4]. Incident response and forensic investigations also gain from this detection process improving security awareness and

knowledge of incidents. Through the identification and response, to port scan attacks organizations take measures to protect their systems, data and assets [15]. Detecting port scan attacks, through machine learning and deep learning entails selecting features and training models to categorize network traffic as either normal or suggestive of a port scan. Approaches involve supervised learning using labeled datasets unsupervised learning for detecting anomalies utilizing learning with neural networks (CNNs) and recurrent neural networks (RNNs) employing ensemble methods that combine different models conducting feature engineering to capture pertinent characteristics and implementing online learning for real time detection [18]. The success hinges on the quality of the training data selecting features and choosing algorithms. Continuous monitoring and assessment are crucial, for maintaining accuracy and flexibility [19].

In our work, the port scans detection is done through the use of feature selection algorithms such as ACO, GA, GWO and machine learning such as SVM, KNN that requires gathering data from network traffic extracting features such, as IP addresses, ports and packet attributes. Then applying the selection features algorithms to select features preparing the data for analysis training a machine learning model like SVM and KNN. Next assessing its effectiveness refining the model for optimal performance and implementing it for real time detection. Ongoing monitoring and updates are essential, for success hinging on high quality training data features selection, appropriate machine learning techniques and continuous performance evaluation.

When using optimization methods, like ACO, GA, GWO and ML for detecting port scans a number of research questions come up; RQ1: How can ACO be applied to enhance feature selection for port detection? RQ2: What advantages does GA offer in terms of feature selection for port scan detection? RQ3: In what ways can GWO improve feature selection for port detection? RQ4: How can machine learning techniques be combined with optimization methods to enhance the detection of port scans? RQ5: What impact does this fusion have on detection performance in feature spaces? In this work, our contributions are: Firstly, Comparative Analysis; we conduct a comparison of feature selection algorithms (such as ACO, GA, GWO) and machine learning algorithms (including SVM, KNN) to assess their accuracy and efficiency, in detecting port scans. This analysis helps identify the algorithm combinations and guides the selection of algorithms for this task. Secondly, Innovative Combinations; the combinations of feature selection and machine learning algorithms to achieve effects that enhance the accuracy and efficiency of port scan detection beyond what individual techniques can offer. So our research lead to solutions that surpass approaches. Thirdly, Evaluation Metrics; Assess evaluation metrics tailored to the specific requirements and challenges of port scan detection. Fourthly, Real world Dataset and Scenarios; our research utilizes a dataset collected from real world network environments that cover port scan activities and normal traffic. This dataset ensures the credibility and applicability of your findings. Moreover, taking into your account real life situations like changing attack strategies and the fluctuating state of networks boosts the usefulness of our method. Finally, enhanced Solutions; when our study results, in precision, productivity or flexibility you help by presenting solutions, for detecting port scans. These solutions give network administrators and security experts' resources to recognize and counter port scan assaults.

The paper is organized in the manner; In Section 2 we mention the existing research, on detecting port scan attack. Section 3 discusses the research subject and the hypothesis formulated for this study. Our methodology is explained in Section 4 and Section 5 describes the experiments carried out in this study. Finally, Section 6 concludes with a discussion of the results.

2 RELATED WORKS

Several researchers have focused on studying the detection of port scan attacks by proposing models to address this issue. For instance, a study by Muhammad Aamir et al., [11] has been focused on utilizing machine learning methods to differentiate between two types of network attacks; port scanning and Distributed Denial of Service

(DDoS) attacks. Researchers compare machine learning algorithms to assess their efficacy, in identifying and distinguishing these attack categories. The research commences by presenting an overview of port scanning and DDoS attacks emphasizing their characteristics and potential impact on network security. It stresses the importance of effective detection mechanisms to counter these cybersecurity threats. Subsequently the authors delve into the selection process of machine learning algorithms for the classification task. They explore a variety of known algorithms such as decision trees, k neighbors (KNN) support vector machines (SVM) and random forests. These algorithms are chosen based on their applicability in handling classification challenges and their track record, in network security contexts. To gauge the performance of these chosen algorithms researchers, utilize a dataset containing diverse network traffic instances encompassing traffic, port scanning attacks and DDoS attacks. They preprocess this dataset by extracting features that capture attributes of these malicious activities. Following data preprocessing experiments are conducted using the dataset to apply each machine learning algorithm for classifying network traffic instances into three categories; traffic, port scanning incidents and DDoS assaults. They assess the algorithms using criteria, like accuracy, precision, recall and F1 score.

Another study by Mir et al., [12], the researchers have been suggested a method that uses intelligence (AI) algorithms to detect and categorize network traffic patterns linked to port scanning activities. The study kicks off by giving an overview of port scanning and its importance, as a step towards network threats. It emphasizes the necessity for prompt detection of port behaviors to boost network security. Following that the authors delve into the methodology utilized for detecting port scans. They explain the process of gathering network traffic data. Preparing it to extract features. These features capture details about network packets, like source and destination ports packet timing and packet size. Subsequently the researchers introduce the AI techniques applied for detecting port scans. They explore algorithms, including machine learning methods like decision trees support vector machines (SVM) and neural networks. These algorithms are trained on the preprocessed dataset to recognize patterns and behaviors associated with port scanning. The study presents the results of experiments conducted using AI algorithms on the dataset. It talks about performance measures used to assess how effective the detection method is, such, as accuracy, precision, recall and F1 score. The authors compare how different algorithms perform and highlight both strengths and weaknesses of each approach. Additionally, the article delves into the real world aspects of putting the suggested port scan detection system into action. It touches upon concerns, like detection, scalability and adjusting to changing attack methods. The writers share their thoughts, on obstacles. Offer solutions to tackle them. Ambedkar et al., [1] focused on the use of machine learning methods to detect probe attacks which're a form of network intrusion where attackers gather information, about a target system without exploiting vulnerabilities. It introduces a technique that utilizes machine learning algorithms to analyze network traffic and recognize patterns linked to probe attacks. By training these algorithms on datasets containing both probe attack data the system can differentiate between malicious incoming network traffic. The study emphasizes the effectiveness of machine learning in identifying probe attacks. Suggests that this approach could bolster network security through accurate threat identification. Another study conducted by Abdulaziz Almazyad et al., [2] introduced an intrusion detection system (IDS) designed to identify probe attacks, in computer networks. Probe attacks are efforts to gather information about a target system without exploiting vulnerabilities. The new IDS combines signature based and anomaly based detection methods to enhance the precision and effectiveness of spotting probe attacks. It utilizes a range of network traffic features, including header details and traffic patterns along with machine learning algorithms to distinguish between normal and malicious network activities. The study demonstrates how the enhanced IDS accurately detects probe attacks while reducing positives. These results indicate that the upgraded IDS plays a role, in bolstering network security by identifying and addressing probe attacks.

Zirak Allaf et al., [3] conducted a study to examine and compare between two methods of side channel attacks, Flush Reload and Prime +Probe on the Advanced Encryption Standard (AES) cryptographic algorithm. These attacks take advantage of the information leaks, from cache access patterns to retrieve keys used in AES. The study utilizes machine learning strategies to scrutinize the gathered cache access patterns and create models for retrieval. By training and assessing machine learning algorithms using the acquired data the research compares how effective Flush Reload and Prime+ Probe attacks are in terms of accuracy, efficiency and resilience. The results of the VOLUME 11, 2023 3 Al-Rewashed et al.: Preparation of Papers for IEEE TRANSACTIONS and JOURNALS study highlight the weaknesses of AES against side channel attacks. Offer insights, into how machine learning techniques can identify and address such threats. In a study [4], the researchers introduced an intrusion detection system (IDS) that utilizes an optimization method known as PSO GWO (Particle Swarm Optimization Grey Wolf Optimization) in conjunction, with Support Vector Machine (SVM). The primary goal of the IDS is to efficiently detect and categorize network intrusions. By employing the PSO GWO algorithm to tune the parameters of the SVM classifier it enhances its performance. Reduces false positives. The system is built on a dataset containing network traffic features with the PSO GWO algorithm playing a role in optimizing the hyper parameters of SVM for classification accuracy. The study showcases how effective the proposed IDS is at identifying types of network intrusions showcasing how the hybrid optimization technique of PSO GWO can significantly boost the SVM classifiers capabilities. These results suggest that this IDS could play a role, in enhancing network security through efficient intrusion detection mechanisms.

Hassan et al., [5] A research paper delving, into the process of discovering and fixing zero day vulnerabilities in software development is the focus here. These vulnerabilities are security loopholes that're unknown to the software company and have not yet been fixed. The paper discusses the challenges in obtaining information about these vulnerabilities in terms of gathering and analyzing data. It delves into the complexities of identifying and addressing zero day vulnerabilities emphasizing the importance of data collection, accurate vulnerability assessment and prompt remediation. The study underscores the significance of taking security measures and fostering collaboration, among researchers, developers and security entities to effectively manage zero day vulnerabilities and bolster software security.

A research study in [6] has been conducted to present a method, for detecting access in wireless sensor networks (WSNs). This method combines the Whale Optimization Algorithm Artificial Bee Colony (WOA ABC) optimization strategy with a developed Convolutional Neural Network (CNN). The study aims to tackle the issue of identifying access in WSNs with resources by fine tuning the CNNs parameters through the WOA ABC algorithm. By leveraging the WOA ABC algorithm, the accuracy and efficiency of the CNN model are improved, allowing it to accurately categorize network data as either normal or malicious. The research illustrates how this combined approach effectively detects types of access in WSNs demonstrating superior performance compared to conventional intrusion detection methods. These results underscore the benefits of integrating optimization algorithms with machine learning techniques to enhance security systems, in sensor networks. Akram Q. M. Algaolahi et al. [7] the research paper delves into the identification of port scanning attacks using machine learning classifiers. Port scanning attacks involve the exploration of a networks ports to uncover vulnerabilities. The paper introduces a technique that leverages machine learning algorithms to scrutinize network traffic and differentiate between activity and signs of a port scanning attack. By training the supervised machine learning classifiers, on datasets labeled with both attack instances the system can discern patterns and characteristics that set benign from malicious network behavior. The study underscores the efficacy of supervised machine learning in pinpointing port scanning attacks. Suggests that this method can bolster network security by enabling early detection and swift response, to such threats. In [8], the research paper delves into evaluating how

well an intrusion detection system (IDS) performs within the realm of the Internet of Things (IoT). The study specifically looks at how different techniques used to select features impact the efficiency of the IDS. Feature selection is, about pinpointing the data variables to enhance how well the IDS works. The research compares methods for feature selection. Examines how they affect both accuracy and computational efficiency when it comes to spotting intrusions in IoT settings. The results emphasize the significance of choosing and tuning features in IoT setups to boost how well intrusion detection systems work reliably. This study adds to our knowledge, about detecting intrusions in systems and sheds light on selecting and using features to enhance detection accuracy.

Sridhar B et al. [9], the extensive research paper offers an overview of how machine learning techniques used to detect attacks, in sensor networks (WSNs). It delves into existing literature showcasing a range of machine learning algorithms and methods employed for intrusion detection in WSNs. The paper also addresses the challenges and constraints linked to WSNs, such as resource limitations, dynamic network structures and the necessity for real time detection. Various attack types including data injection, node compromise and routing attacks are discussed alongside how machine learning strategies can identify and counter these threats. The study evaluates the strengths and weaknesses of machine learning algorithms in detecting WSN attacks while suggesting avenues for future research, in this domain. In essence this survey sheds light on the role of machine learning in safeguarding wireless sensor networks from attack scenarios. In [10], a research paper delving, into the investigation of attacks on a network attack detection system utilizing the Random Forest algorithm. Adversarial attacks involve efforts to manipulate or deceive machine learning models. The paper scrutinizes the susceptibility of Random Forest based network attack detection systems to attacks and delves into attack strategies. The study examines how adversarial attacks impact the accuracy of the detection system and assesses defense mechanisms effectiveness. The research outcomes shed light on the weaknesses of Random Forest based detection systems. Suggest measures to bolster their resilience against adversarial attacks. This study enhances our comprehension of security challenges, in network attack detection systems. Underscores the significance of developing models capable of withstanding adversarial interference.

Table1: Datasets for Intrusion Detection

Datasets	year	information
KDD 99 CUP	1999	41 Features represent the legitimate and attack traffic
CAID 07	2007	Containing the flooding traffic of SYN, ICMP,HTTP protocols
CAID 08	2008	Legitimate and attack traced monitored of Chicago and san Jose
NSL-KDD	2009	Refined version of KDD99dataset after removal of duplicate records
ISCX	2012	Traffic from real world physical test environment
UNSW-NB15	2015	49 features covering 9 types of attacks
CICIDS2(our dataset) 017	2017	78 features with normal traffic and attacks

3 RESEARCH METHODOLOGY

To detect port attacks using feature selection techniques and machine learning (ML) algorithms we use the following process involves key steps; Firstly, data Collection; Gather a dataset that includes network traffic data, with both traffic

instances and port scan attacks. Secondly, feature Selection; Employ feature selection algorithms to pinpoint features that differentiate between traffic and port scan activities. Thirdly, feature Extraction; Extract the selected features from the preprocessed dataset, such as IP addresses, ports, timing, and packet size or protocol information. Fourthly, ML Algorithm Selection. Fifthly, training; Train the ML algorithms using the extracted features along with labeled data to recognize patterns associated with port attacks. After that in step six, the evaluation and Validation; assess the trained models using metrics such, as accuracy, precision, recall, F1 score or AUC ROC. Validate these models using datasets to evaluate their generalization capabilities.

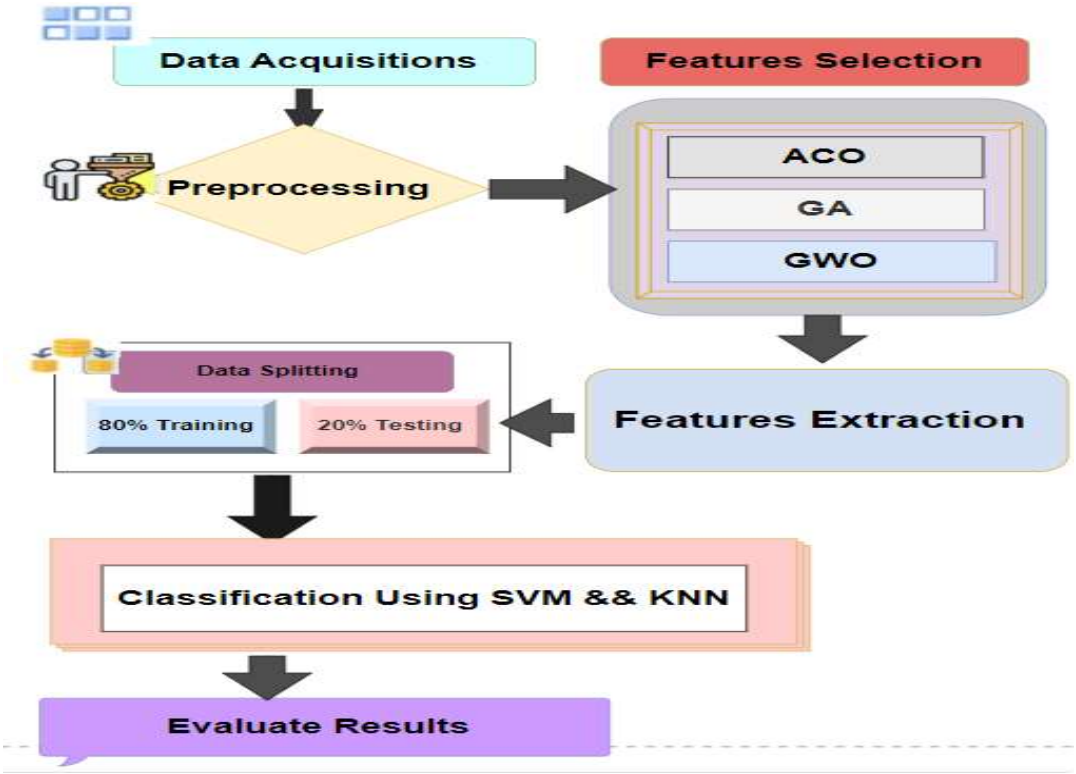

Figure 2: Research Methodology Framework

3.1 Data Acquisition Description:

In the realm of safeguarding against changing network threats, Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) play a role. Nevertheless, the efficacy of anomaly based intrusion detection methods is hindered by the

absence of testing and validation datasets. An evaluation of eleven datasets spanning from 1998 reveals that many are outdated and unreliable lacking in diversity and failing to cover a spectrum of recognized attacks or anonymize payload data. In contrast the CICIDS2017 dataset shines for its inclusion of traffic data and typical attack scenarios mirroring real world network activities closely. It furnishes labeled flows, with timestamps, source/destination IPs, ports, protocols and attack types stored in CSV files. Moreover, this dataset emphasizes generating background traffic via the B Profile system to replicate human interaction patterns. It scrutinizes the actions of 25 users across protocols while simulating attacks like Brute Force FTP and SSH strikes, Denial of Service (DoS) assaults, Web Attacks, infiltration endeavors and Botnet operations. The dataset fulfills eleven criteria for crafting a benchmark dataset which encompasses a network configuration and various sources of traffic data. During attack simulations network traffic data, as memory dumps and system call information were collected from compromised machines. Over 80 different network flow characteristics were obtained using the CICFlow Meter tool and the data set generated is available, in CSV format. https://www.kaggle.com/code/saqlainhussainshah/cicids-2017-knn)

3.2 Features Selection:

Feature selection methods play a role, in detecting port scan attacks by identifying the relevant features. These methods automate the selection of features that effectively distinguish network traffic from port scanning activities. There are many methods such as mutual Information; this assesses the relationship between variables to determine feature importance based on their information content. Secondly, Information Gain; It measures how much uncertainty in the target variable is reduced by considering each features contribution. Thirdly, Chi Square Test; this evaluates the independence between variables to understand how features are associated with the target variable. Fourthly, Recursive Feature Elimination (RFE); It systematically eliminates features based on a specific criterion until a desired subset is achieved. Six, Wrapper Methods; These assess machine learning algorithm performance using feature subsets utilizing search algorithms to find the subset. Seventhly, L1 Regularization (Lasso); by penalizing weights this method promotes sparsity in the feature space. Selects features with non-zero coefficients. Eighthly, Correlation based Feature Selection; this identifies relationships between features and the target variable, selecting those with correlations while discarding redundant ones. Choosing an algorithm depends on factors like characteristics feature count and the desired tradeoff between performance and efficiency. In our work, we use three algorithms for this purpose (GA, ACO, and GWO)

1. Genetic Algorithm (GA)

Genetic algorithms, known as GAs play a role, in selecting features for detecting port scans. These algorithms use selection and evolution to find the subset of features. The process includes the following steps:

- Initialization; Creating a group of feature subsets.
- Assessing the fitness of each subset by measuring its effectiveness in distinguishing traffic from port activities.
- Selection; Choosing candidate subsets for reproduction based on their fitness mimicking the idea of "survival of the fittest."
- Genetic Operators;

- Crossover; Exchanging features between selected subsets to create subsets with diversity.
- Mutation; Randomly changing features in subsets to increase population diversity.

- Evaluating the fitness of created offspring subsets.
- Replacement; Swapping some candidate subsets with offspring subsets based on their fitness levels to promote performing solutions.
- Continuing selection, crossover and mutation for a number of generations or until a termination condition is met.
- Choosing the subset with the fitness (. Meeting a desired threshold) as the final feature subset, for port scan detection.

In essence genetic algorithms navigate through feature combinations to improve port detection capabilities by evolving groups of potential solutions. To enhance port detection efficiency, the focus is, on assessing fitness choosing candidates and using genetic techniques to identify the most effective or nearly optimal subsets.

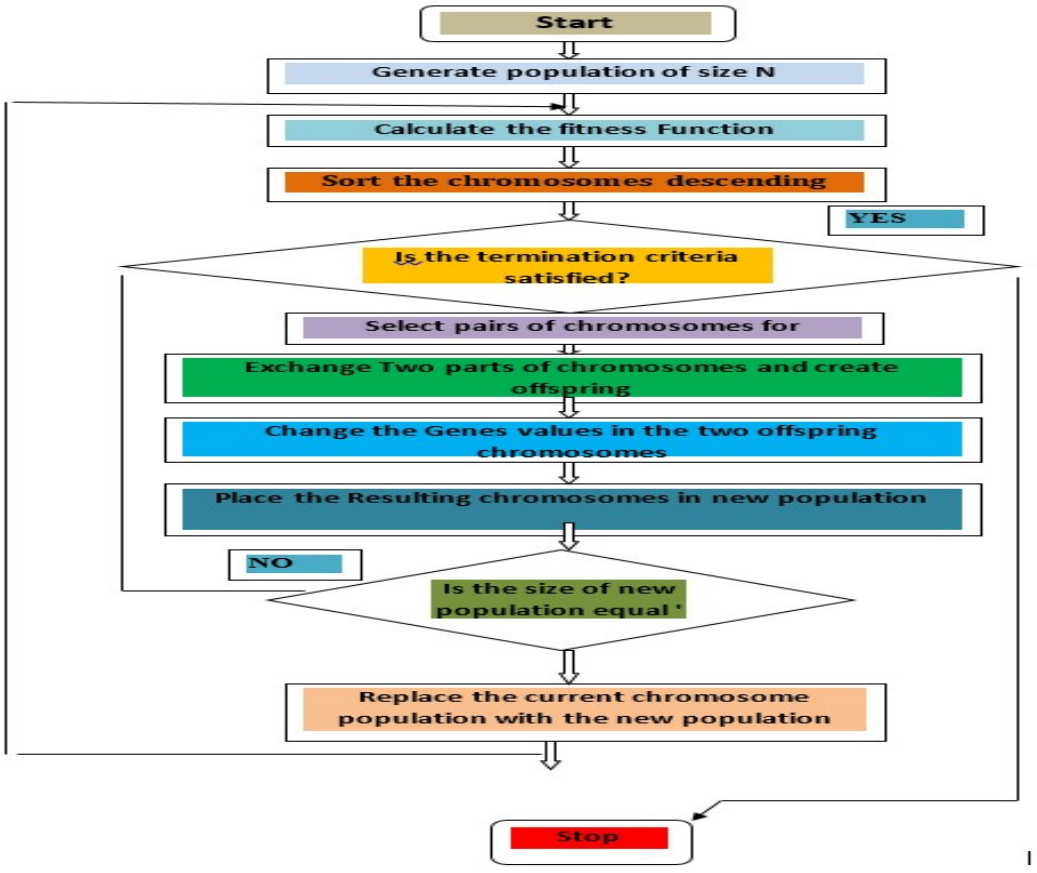

Figure 3: Genetic Algorithm Flowchart

The parameters used in our GA an algorithm (GA); Firstly. The population size parameter decides the number of individuals (candidate solutions) in each generation of the GA. Having a population size allows for an exploration of the search space though it may lead to increased computational complexity. Secondly, Maximum Generations. This parameter sets the number of iterations or generations that the GA will go through before stopping. It serves as a criterion, for ending the algorithm. Helps manage the runtime of the GA. Thirdly, Crossover Rate. The crossover rate parameter determines how likely it is for two parent individuals to exchange information during crossover. Crossover mixes material from two parents to create offspring with a blend of their traits. Fourthly, Mutation Rate and fifthly, the mutation rate parameter defines the likelihood of a gene (or feature) in an individual undergoing mutation. Mutation introduces changes into an individual's makeup to encourage exploration in the search space. Sixty, Selection Mechanism; Seventhly, this parameter dictates how individuals are chosen as parents for reproduction in the GA. Different selection methods like tournament selection, roulette wheel selection or rank based selection can be used to pick individuals with fitness levels, for reproduction. Eighthly, the elitism setting decides whether the performing individuals, from one generation are kept unchanged in the generation. The termination criteria setting establishes the conditions that signal when the GA should conclude its operations. Typical termination criteria involve reaching a limit, on the number of generations achieving a solution or not seeing substantial enhancements after a specific number of generations.

Table 2: Parameters Setting of GA

Parameter	Default Value Range
Population Size	100-200
Maximum Generations	100-1000
Crossover Rate	0.6-0.9
Mutation Rate	0.01-0.05
Selection Mechanism	-
Elitism	1
Termination Criteria	-

2. Ant Colony Algorithm (ACO)

Ant Colony Optimization (ACO) is a strategy inspired by how ants search for food often used to tackle optimization issues. Besides its use, in solving problems ACO can also be applied to select features in detecting port scans. The steps of how ACO works for feature selection;

- Getting Started;
- Begin by forming a group of ants with each ant representing a solution (a set of features).

- Set up the pheromone trail matrix indicating the desirability of each feature.

Ant Actions;

- Each ant navigates through the feature space to create a solution step by step.
- At every stage an ant selects features probabilistically for its set based on the intensity of pheromone trails and a heuristic function (such, as feature importance or relevance).
- The ant continues adding features until certain conditions are met.
- Updating Pheromones;
- Once an ant completes its solution adjust the pheromone trail.
- Enhance the attractiveness of selected features by increasing their pheromone intensities.
- Reduce the appeal of selected features by lowering their pheromone intensities. Conclusion; the process of movement continues for a number of iterations or until a specific condition is met, such as convergence or reaching the maximum number of iterations.
- Feature Subset Selection;

Upon completion of the algorithm the optimal feature subset is chosen based on the strength of the pheromone trails. Features, with pheromone trail strengths are deemed significant and included in the final subset. In using ACO for feature selection in port detection the ants are guided through the feature space by leveraging pheromone trail strengths. As the ants build solutions iteratively adjustments are made to enhance the attractiveness of features that aid, in distinguishing between traffic and port scan activities effectively.

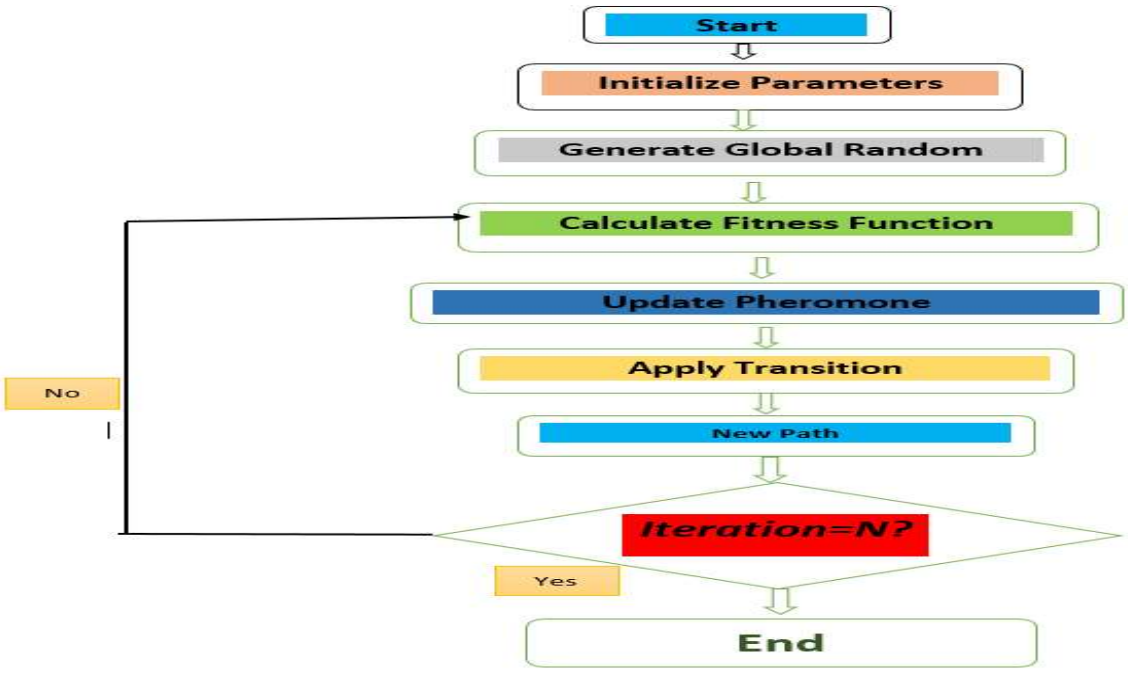

Figure 4: Ant Colony Algorithm Flowchart

The Parameters that used in our Ant Colony Optimization (ACO); firstly, Number of Ants. This parameter sets the quantity of ants, within the group. Each ant symbolizes a solution or route in the issue at hand collectively exploring the search space. Secondly, Number of Iterations; this parameter defines the number of times the ACO algorithm will operate. It determines how frequently ants will travel through the problem area leaving pheromones and adjusting solutions. Thirdly, Pheromone Decay Rate; This parameter dictates how pheromone trails laid by ants fade, across iterations. Decay is essential to prevent solution stagnation and promote exploration of routes. Fourthly, Pheromone Intensity; this parameter signifies the quantity of pheromones left on paths. It impacts path attractiveness during exploration, where higher intensity denotes appeal. Fifthly, Alpha; the alpha factor is a weight that influences how much importance pheromone trails hold in decision making. It regulates how much influence pheromone details have on behavior with higher values underscoring trail significance. sixty, Beta; The beta value plays a role, in guiding decision making by determining how much weight heuristic information, such as distance and cost carries. It regulates how impact heuristic information has on behavior with higher values highlighting its significance. Seventhly, Exploration Factor; the exploration factor determines the likelihood of ants opting for exploration, than exploitation. It enables a blend of exploring paths and exploiting well known paths in the optimization phase.

Table 3: Parameters Setting of ACO

Parameter	Default Value Range
Number of Ants	10-100
Number of Iterations	100-1000
Pheromone Decay Rate	0.1-0.9
Pheromone Intensity	-
Alpha	1-5
Beta	1-5
Exploration Factor	0.1-0.9

3. Gray Wolf Optimization Algorithm (GWO)

The Gray Wolf Optimization (GWO) algorithm draws inspiration from the structure. Hunting tactics of gray wolves. While GWO is commonly utilized for optimization tasks it can also be tailored for selecting features in detecting port attacks. The steps of GWO are:

- Set the population size, iteration count and other key parameters.
- Form a group of wolves with each wolf symbolizing a potential set of features.
- Randomly position the wolves in the feature space.
- Hunting Phase; Each wolf adjusts its position based on wolves, in the pack.
- The alpha, beta and delta wolves serve as leaders with fitness levels.

- Update each wolfs position using equations inspired by wolf hunting behaviors that balance exploration and exploitation.
- Assessing Fitness;
- Evaluate the fitness level of each wolf within the population.
- The fitness function gauges how well a feature subset distinguishes, between network traffic and port attacks.
- Leader Identification;
- Determine the alpha, beta and delta wolves based on their fitness scores.
- These leading wolves represent the feature subsets identified so far. Exploring and Utilizing Resources;
- Adjust the wolves' positions by using equations that consider both exploring solutions and improving existing ones.
- The adjustments are made based on the positions of the alpha, beta and delta wolves who act as leaders.
- Completion;
- Continue until a specific termination condition is reached (e.g., convergence or reaching the iteration limit).
- Selecting Key Features;
- Once the algorithm completes its iterations choose the optimal feature subset by evaluating the wolves' fitness values.

Typically, the feature subset linked to the alpha wolf (the leader, with the fitness) is chosen as the solution. In GWO wolves' positions in the feature space are updated over time with leaders guiding others movements. The goal is to maintain a balance between exploring features and refining existing ones to identify near optimal solutions, for detecting port scan attacks.

Figure 5: Gray Wolf Algorithm Flowchart

The parameters typically utilized in OUR Gray Wolf Optimization (GWO); Firstly, Population Size; The population size factor dictates the quantity of wolves within the group. Each wolf serves as a solution, within the problem domain collectively navigating through the search space. Secondly, Maximum Number of Generations; Thirdly, the maximum number of generations parameter establishes the limit on iterations or generations for the GWO algorithm. It sets boundaries on how times wolves will adjust their positions and seek solutions. Fourthly, Crossover Rate; the crossover rate parameter influences the likelihood of two wolves sharing information during crossover operations. Crossover serves as a mechanism merging details from two wolves to generate offspring with a blend of their attributes. Fifthly, Mutation Rate; the mutation rate parameter governs the chance of material in a wolf undergoing mutation. Mutations introduce alterations into wolves' genetic makeup enabling exploration within the search space. Sixty, Exploration Factor; The exploration factor aspect acts as a regulator determining the equilibrium between exploration and exploitation, in GWO methodology. It impacts decision making among wolves defining the extent to which they explore solutions versus exploiting existing ones. Seventhly, Termination Criteria; The termination criteria parameter outlines the circumstances that dictate when the GWO algorithm should cease its operation. Typical termination criteria involve reaching a limit, on the number of generations discovering a solution or failing to achieve substantial enhancements after a specific number of iterations.

3.3 Features Classification Using ML algorithms (SVM &&KNN)

Support Vector Machines (SVM) and k Nearest Neighbors (KNN) are two algorithms commonly utilized in identifying port attacks. In Support Vector Machines (SVM); SVM, a machine learning technique mainly used for classification purposes, like intrusion detection can be trained to differentiate between network activity and suspicious port scanning behavior. By establishing a hyperplane that effectively separates data point classes within the feature space SVM can recognize patterns and traits linked to port scan attacks. Through training on labeled data indicating network traffic or port scans SVM constructs a model to classify instances as either regular or part of a port scan attempt. Regarding to k Nearest Neighbors (KNN); KNN, a straightforward machine learning method employed for classification and regression tasks is also applicable, in identifying port activities by categorizing network traffic as benign or indicative of a scan. Utilizing the majority class from the k neighbors surrounding a data point in the feature space enables KNN to make predictions effectively. To detect port scans using KNN you need a training set that includes labeled instances of port scan traffic. When faced with an instance KNN measures the distances, between the instance and its k nearest neighbors in the feature space. It then assigns a class label based on what the majority of those neighbors suggest. SVM and KNN both have their pros and cons when it comes to spotting port attacks; SVM shines in feature spaces by finding intricate decision boundaries. This strength is especially handy when dealing with data that's not easily separable in a fashion. On the hand KNN is an algorithm to grasp. It works well where decision boundaries are nonlinear and the feature space isn't overly extensive. Deciding between SVM and KNN for detecting port scan attacks depends on factors, like characteristics, decision boundary complexity and available computational power.

4 RESULTS AND DISCUSSION

Assessment tools are utilized to gauge how machine learning models or algorithms perform.

When it comes to classification assignments the evaluation metrics commonly employed are; Accuracy is a used measure to assess how correct a classifiers predictions are, across fields. It calculates the ratio of predicted instances (including positives and true negatives) to the total instances considered. While accuracy provides an evaluation of the models performance it can be misleading when there is a distribution of classes in the dataset. Sensitivity, also called recall or true positive rate indicates the percentage of instances correctly identified by the classifier. This metric is calculated by dividing the number of positives by the sum of positives and false negatives. Sensitivity is particularly useful when the goal is to minimize negatives like in detecting all cases of a disease. Specificity measures how negative instances are identified by the classifier. It calculates the percentage of negatives out of the sum of negatives and false positives. Specificity becomes important when reducing positives is crucial such, as distinguishing individuals without a disease effectively. Precision or positive predictive value evaluates how many predicted instances are actually positive. When determining precision, it involves calculating the ratio of positives to the sum of positives and false positives. Precision provides insights, into how the classifier can avoid positives, which is especially useful in scenarios where the cost of false positives is significant. The F measure, also known as the F1 score combines precision. Recall to create a metric that balances both aspects of the classifiers performance. This metric calculates the mean of precision and recall providing a value to assess how effective the classifier is overall. The F measure is beneficial, in situations where classes are unevenly distributed or when there's a need to reduce both positives and false negatives at the time. The equations related to each our metrics are as follow:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

$$\text{Sensitivity} = (TP) / (TP + FN)$$

$$\text{Specificity} = (TN) / (TN + FP)$$

$$\text{Precision} = TP / (TP + FP)$$

$$\text{F1-score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

	ACO+SVM	ACO+KNN	GA+SVM	GA+KNN	GWO+SVM	GWO+KNN
Accuracy	0	0.998	0.998	0.997	0.999	0.997
Sensitivity	0.995	0.999	0.999	0.999	0.999	0.997
Specificity	0.996	0.999	0.999	0.998	0.999	0.997
Precision	0.995	0.997	0.998	0.996	0.999	0.997
Recall	0.996	0.999	0.999	0.998	0.999	0.997
F-measure	0.998	0.998	0.999	0.997	0.999	0.997

Figure 6: Results Of Our Proposed Models

Figure 6 show that all the techniques show accuracy levels, ranging from 99.7%, to 99.9% indicating overall classification performance. Combinations like ACO+SVM, ACO+KNN, GA+SVM and GWO+SVM consistently deliver results across all evaluation criteria showcasing performance in sensitivity, specificity, precision, recall and F measure. While GA+KNN and GWO+KNN also perform across these metrics their scores are slightly lower compared to the methods mentioned earlier. In general, GWO+SVM stands out with top notch scores, in accuracy, sensitivity, specificity, precision, recall and F measure among the methods listed.

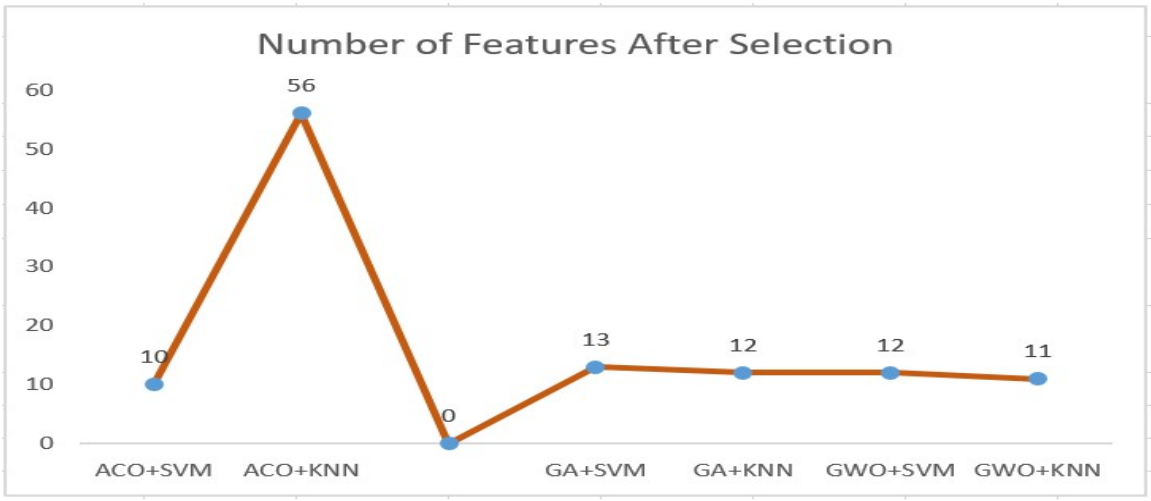

Figure 7: Number Of Features After Selection For Each Method

Figure 7 show that the ACO+SVM (10 features), GWO+SVM (12 features); both of these methods use feature selection, alongside SVM for categorization purposes. ACO+SVM algorithm for a set of features possibly honing in on the crucial ones. On the hand GWO+SVM goes for a larger selection of features offering a more comprehensive view of the data. ACO+KNN (56 features) and GA+SVM (13 features); ACO+KNN goes for the feature set indicating a thorough exploration of the feature space. In contrast GA+SVM chooses features than ACO+KNN but more than ACO+SVM and GWO+KNN.GA+KNN (12 features) and GWO+KNN (11 features); Both methods use feature selection in conjunction with KNN for classification purposes. GA+KNN and GWO+KNN both opt for a number of features suggesting they focus on informative aspects. The choice between these methods depends on factors such as specifics, available computational resources and the balance between feature complexity and classification accuracy. Evaluating each method using metrics is essential to determine the approach, for a given classification task.

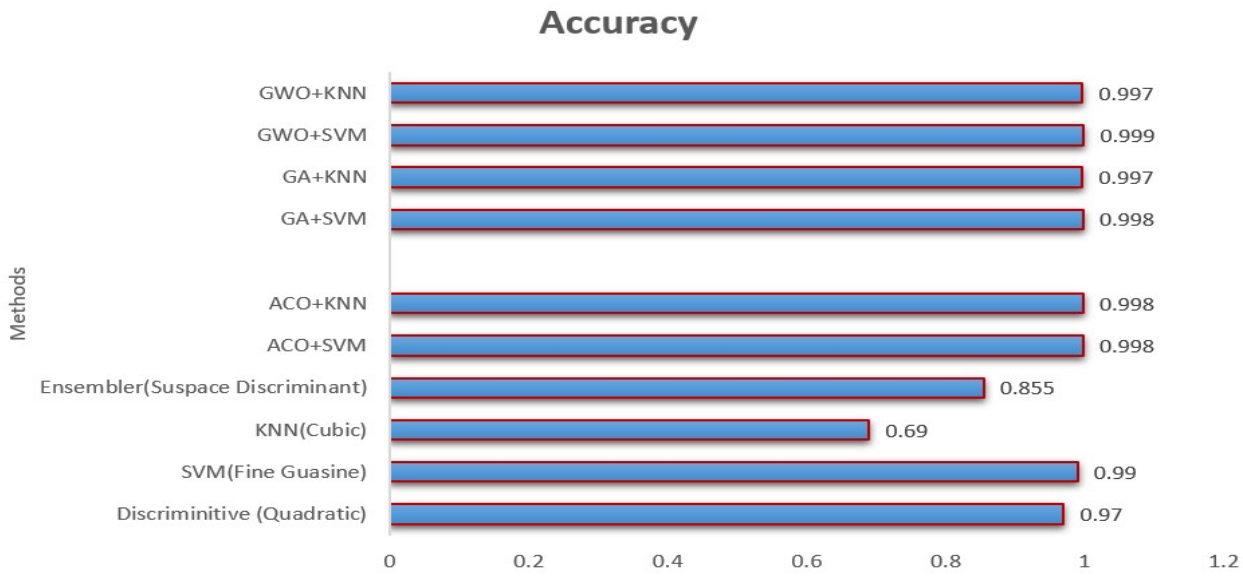

Figure 8: Comparing between our work and previous works

Figures 8 to 14 provide the comparison between the machine learning methods; Support Vector Machine (Fine Gaussian) Ant Colony Optimization + SVM Genetic Algorithm + SVM and Grey Wolf Optimization + SVM show accuracy levels ranging from 0.998, to 0.999.Ant Colony Optimization + K Nearest Neighbors, Genetic Algorithm + K Nearest Neighbors and Grey Wolf Optimization + K Nearest Neighbors also display accuracy rates varying from 0.997 to 0.998.The Discriminative (Quadratic) and Ensembles (Subspace Discriminant) techniques achieve accuracies of 0.97 and 0.855 correspondingly. Among the listed methods K Nearest Neighbors (Cubic) performs the least accurately with a score of 0.69.In terms of accuracy Grey Wolf Optimization + SVM and Grey Wolf Optimization + K Nearest Neighbors stand out as the accurate methods, with scores of 0.999 and 0.997 respectively. Support Vector Machine (Fine Gaussian) Ant Colony Optimization + SVM Genetic Algorithm + SVM Ant Colony Optimization + K Nearest Neighbors and Genetic Algorithm + K Nearest Neighbors also demonstrate accuracy rates above 0.997.The results show that our methods outperform the previous methods.

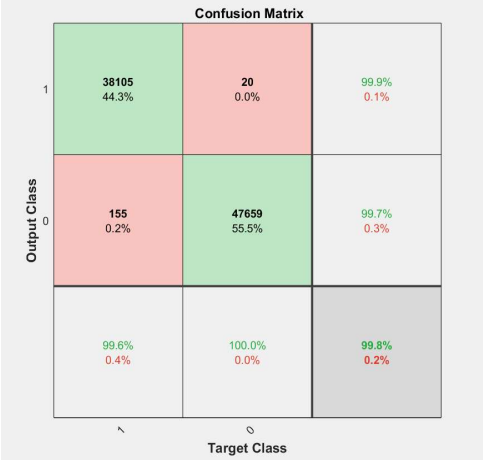

Figure 9: Evaluation results for ACO with KNN

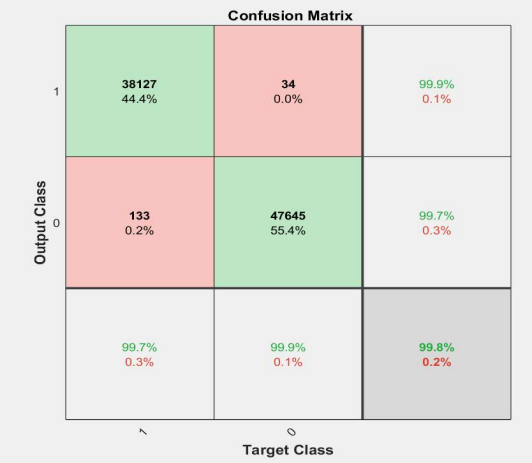

Figure 10: Evaluation results for ACO with SVM

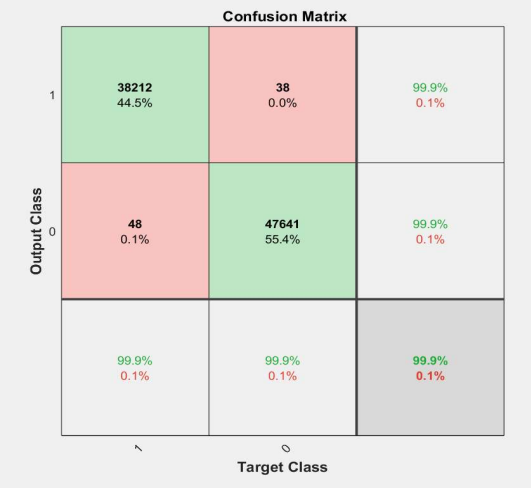

Figure 11: Evaluation results for GA with KNN

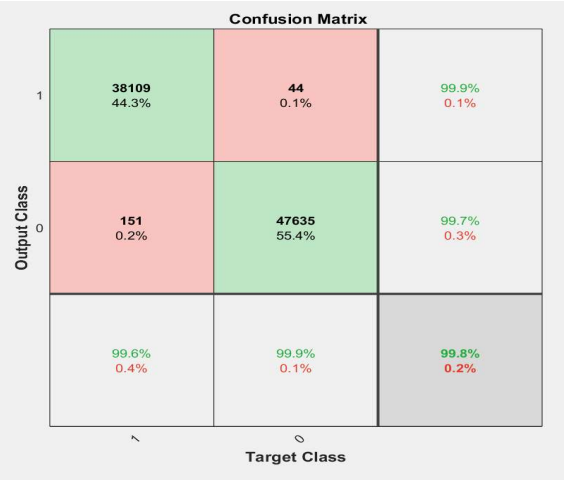

Figure 12: Evaluation results for GA with SVM

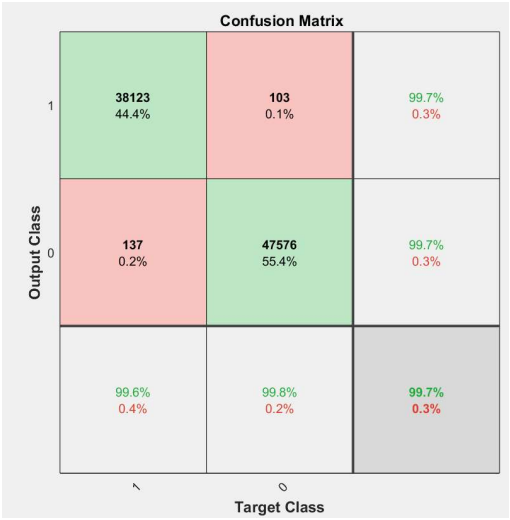

Figure 13: Evaluation results for GWO with KNN

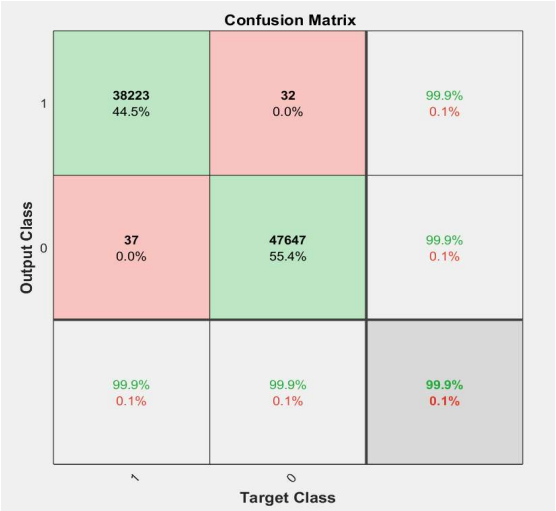

Figure 14: Evaluation results for GWO with SVM

5 CONCLUSIONS

Port scanning attacks have uses, such, as mapping out network structures, pinpointing services or gearing up for additional exploits. It's worth mentioning that port scans aren't always harmful. This can serve as a tool, for system admins and cybersecurity experts to spot and fortify weaknesses in their systems. To defend against port attacks companies frequently deploy firewalls, intrusion detection systems (IDS) and network monitoring tools to catch and prevent malicious port scanning actions. Detecting port scan attacks is essential, for staying of threats evaluating vulnerabilities responding to incidents and safeguarding network security. This practice allows companies to pinpoint points react swiftly to attacks and put in place the right security protocols leading to a lower chance of cyber-attacks being successful. In our work, we start by Collecting Data; Start by gathering network traffic data that contains information. Then extracting features. Extract features from the collected data to train the machine learning model effectively. Next, choosing Features; Utilize algorithms to select the informative features such as ant colony algorithm (ACO), genetic algorithm (GA), and gray wolf optimization (GWO) which helps in reducing complexity. Additionally, training machine learning models; use the selected features to train a machine learning model. Then we use classification operation for the features using machine learning algorithms such as support vector machine (SVM), and nearest neighbor (KNN). Finally, assessing Model Performance; evaluate how well the model performs using metrics, like precision, recall, F1 score or accuracy. Our results show that we achieves good results reaching to over 99% for all proposed models. In conclusion, detecting port scans is crucial, for boosting network security as it helps spot and counter port attacks. Through the use of feature selection algorithms and Machine Learning (ML) methods, can help researchers to identify port behaviors and attacks in more efficiently.

Acknowledgement: This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia (Grant No. KFU241590).

REFERENCES

[1] A. I. Saleh, F. M. Talaat, and L. M. Labib, “A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers,” *Artif. Intell. Rev.*, vol. 51, pp. 403–443, 2019.

[2] S. Velliangiri and P. Karthikeyan, “Hybrid optimization scheme for intrusion detection using considerable feature selection,” *Neural Comput. Appl.*, vol. 32, pp. 7925–7939, 2020.

[3] Q. M. Alzubi, M. Anbar, Z. N. M. Alqattan, M. A. Al-Betar, and R. Abdullah, “Intrusion detection system based on a modified binary grey wolf optimisation,” *Neural Comput. Appl.*, vol. 32, pp. 6125–6137, 2020.

[4] D. Arivudainambi, V. K. KA, and S. Sibi Chakkaravarthy, “LION IDS: A meta-heuristics approach to detect DDoS attacks against Software-Defined Networks,” *Neural Comput. Appl.*, vol. 31, pp. 1491–1501, 2019.

[5] M. Singh, M. Singh, and S. Kaur, “Issues and challenges in DNS based botnet detection: A survey,” *Comput. \& Secur.*, vol. 86, pp. 28–52, 2019, doi: https://doi.org/10.1016/j.cose.2019.05.019.

[6] A. E. Ezugwu, J. O. Agushaka, L. Abualigah, S. Mirjalili, and A. H. Gandomi, “Prairie dog optimization algorithm,” *Neural Comput. Appl.*, vol. 34, no. 22, pp. 20017–20065, 2022.

[7] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, “Slime mould algorithm: A new method for stochastic optimization,” *Futur. Gener. Comput. Syst.*, vol. 111, pp. 300–323, 2020.

[8] DARPA, AFRL/RI, and K.-99 Organizers, “KDD Cup 1999 Dataset,” *KDD-99 Fifth Int. Conf. Knowl. Discov. Data Min.*, 1999, [Online]. Available:

https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data

[9] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "NSL-KDD Dataset," *Univ. New Brunswick, Can. Inst. Cybersecurity*, 2009, [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html

[10] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, "Kyoto 2006+ Dataset," *Kyoto Univ.*, 2006, [Online]. Available: http://dx.doi.org/10.23721/100/1478781

[11] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri, and M. Rida, "A hybrid optimization framework based on genetic algorithm and simulated annealing algorithm to enhance performance of anomaly network intrusion detection system based on BP neural network," in *2018 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, 2018, pp. 1–6.

[12] P. Amudha, S. Karthik, and S. Sivakumari, "A hybrid swarm intelligence algorithm for intrusion detection using significant features," *Sci. World J.*, vol. 2015, 2015.

[13] S. M. H. Bamakan, B. Amiri, M. Mirzabagheri, and Y. Shi, "A new intrusion detection approach using PSO based multiple criteria linear programming," *Procedia Comput. Sci.*, vol. 55, pp. 231–237, 2015.

[14] G. J. Mejtsky, "A metaheuristic algorithm for simultaneous simulation optimization and applications to traveling salesman and job shop scheduling with due dates," in *2007 Winter Simulation Conference*, 2007, pp. 1835–1843.

[15] A. S. Eesa, Z. Orman, and A. M. A. Brifcani, "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2670–2679, 2015.

[16] B. Hajimirzaei and N. J. Navimipour, "Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm," *Ict Express*, vol. 5, no. 1, pp. 56–59, 2019.

[17] K. Selvakumar *et al.*, "Intelligent temporal classification and fuzzy rough set-based feature selection algorithm for intrusion detection system in WSNs," *Inf. Sci. (Ny).*, vol. 497, pp. 77–90, 2019.