DOI: 10.15849/IJASCA.250330.13

Int. J. Advance Soft Compu. Appl, Vol. 17, No. 1, March 2025
Print ISSN: 2710-1274, Online ISSN: 2074-8523

Analysis and Comparison of Raw Network Packet Datasets Using Machine Learning Classification and Grey Wolf Optimization

Hussein A. Al-Ofeishat[1] and Jawdat S. Alkasassbeh[2] and Albara Awajan[3] and Moutaz Alazab[3,4]

[1]Computer Engineering Department,
Al-Balqa Applied University, Al-Salt, Jordan; Ofeishat@bau.edu.jo
[2]Department of Electrical Engineering, Faculty of Engineering Technology,
Al-Balqa Applied University, Amman, Jordan
Jawdat1983@bau.edu.jo
[3]Intelligent Systems Department,
Al-Balqa Applied University, Al-Salt, Jordan;
a.wajan@bau.edu.jo
[4]Cybersecurity Department, School of Computing and Data Sciences
Oryx Universal College, Liverpool John Moores University
Doha, Qatar
m.alazab@bau.edu.jo

Abstract

A Machine Learning (ML)- Driven automatic defense mechanism has become an efficient way to safeguard organizations from massive volumes of intrusions executed in random patterns. A successful defense against a cyber-attack reveals the ineffective nature of the intrusion on the target network, which guides the intruders in modifying their methods. This ever-evolving dynamics of cyber-attacks necessitate frequent analysis of the effectiveness of the existing ML models against different types of intrusions. This paper presents an exploratory analysis of the effectiveness of ten ML models in various settings to defend against cyber-attacks performed using the UNSW-NB15 dataset. This analysis involves an innovative integration of the Grey Wolf Optimization (GWO) algorithm to reduce the feature vector of the dataset by using the features with strong correlation with the target variables. The exploratory analysis of this study finds the Random Forest and Extra Trees models to be the most accurate, with 97.68% and 97.53%, respectively, along with the Fact sheet showing an increase in the performance ratio of the Random Forest model reaching a very high level of 98.25%, followed by the success rate of Extra Trees model which reached 98.17%. GWO efficiently reduces the feature set from 39 to 20 to improve the model performance and reduce computational time. The findings of this study lay the groundwork for researchers and developers intending to apply ML models to defend against cyber-attacks automatically.

Keywords: *UNSW-NB15; Machine Learning; Random Forest; Classification; Raw network packets.*

Received 5 January 2025; Accepted 10 March 2025

1 Introduction

In an era of breakneck technological progress and an all-encompassing online presence, maintaining the integrity and confidentiality of stored data is one of the primary issues for both a person and a business. Rapid changes in technology come with a variety of new security threats [1]. The cases of cyber-attacks and malware proliferation have notably grown over recent years. Biggest Misconceptions - Cybersecurity vs Information Security It is imperative to know/understand the difference between "Cybersecurity" and "Information Security," as Cybersecurity covers a broader range of issues besides traditional IT security. Because of digitization, advanced cyber security is necessary to protect data and digital systems from unauthorized access [2].

That is why the term Cyber Threat Intelligence (CTI) has emerged.

CTI is a method of systematically analyzing cyber-attacks using AI. The analytical results are later used to develop strategies for mitigating cyber-attacks. Accurately identifying an attack's background is very easy with CTI. However, data collection and analysis for CTI is very challenging. The challenges related to high-dimensional data are extensive and affect a range of disciplines, including cybersecurity. Another challenge is the availability of various methods utilized by cybercriminals [3]. Traditional security methods are different from modern cybersecurity. The traditional approaches have limitations. They struggle to defend against zero-day vulnerabilities, unpatched software attacks, and social engineering. That is why modern CTI is essential [4]. It involves ML approaches. ML methods are crucial in advancing cybersecurity, surpassing older rule-based systems. ML algorithms can analyze data, identify patterns, and detect anomalies efficiently. They use supervised, unsupervised, and reinforcement learning to learn from past incidents, predict future threats and adapt to new attacks. This adaptability is essential to defend against ever-evolving cyberattacks. However, ML approaches suffer from the challenge of lacking high-quality training data. The UNSW-NB15 dataset has been prepared to address this challenge. It was prepared using network simulation and comprehensive traffic data [5,6]. This paper delves into the ethical and legal obstacles associated with the integration of machine learning in cybersecurity, going beyond the purely technological issues. The more ML algorithms are integrated within security frameworks, the more consequential privacy, data protection, and algorithmic bias issues become as the machine learning algorithms process large volumes of data, which may contain sensitive data being exposed. For the public to continue to trust ML models and for organizations to adhere to legal standards, it is vital to ensure that the entire lifecycle of the development and deployment of ML models is done with responsibility in mind. This work adds to the growing conversation about ethical AI in cybersecurity and underscores the need for transparency, accountability, and fairness in the creation of ML security products.

We organized our paper as follows: Section 2 provides the background and related work on Cyber Security and Machine Learning. The following section, Section 3, is a discussion of the methodology used in conducting the research investigation, comprising data collection, pre-process, and machine learning algorithms. In section 4, experimental results are shown and summarized. Section 5 provides the conclusions and insights for future research and the potential implications of our findings. As a result, this introduction to cybersecurity with machine learning breaks down into the following distinct sections, with which we are confident that readers would be well

versed with the challenges, methodologies, and implications of incorporating machine learning approaches into cybersecurity solutions.

1.1 Contributions

The study contributes to state-of-the-art network intrusion detection based on machine learning and optimization. The main contributions of this work are:

(A) Grey Wolf Optimization (GWO) for Feature Selection in the Research:

The research did a fantastic job integrating the GWO for feature selection and reducing the features from a file of 39 features to 19 features. This improvement significantly enhanced both classification accuracy and the running time of the machine learning models employed in this study.

(B) Model Comparison:

The performance of various machine learning algorithms, like Random Forest, Extra Trees, LSTM, GRU, MLP, etc., was compared in detail. Model performance on the UNSW-NB15 Dataset was measured using evaluative metrics such as accuracy, recall, precision, and F1-Score. The Random Forest and Extra Trees models had the highest accuracy at 97.68% and 97.53%, respectively, showing the success of ensemble models for raw network packet data classification.

(C) Integration of Sequential Data Processing Models:

The study used LSTM and GRU models in the processing of temporal data extracted from network traffic, proving to be efficient in temporal dependency modeling and obtaining superior accuracies in packet classification. These approaches could be potentially used together to model both time-wise behaviour and sequence interactions of multiple packets, thus reducing overall error rates. Meanwhile, with respective accuracy rates of 96.48% and 96.33%, STM and its counterpart GRU are confirmed to be good solutions for models to deal with sequential data in network traffic.

(D) Practical Implications for Network Security:

With a view to improving network security in practice, the evidence obtained in this study has practical implications. The approach can effectively enhance the reliability of intrusion detection systems by identifying and categorizing network threats with high precision.

2 Literature Review

Supervised learning approaches, including decision trees, SVMs, and ANNs [8,9], have been employed for years to solve several real-life time problems, but the challenge of rare assault detection remains largely unresolved as they are susceptible to a high false-positive ratio. In this work, we introduce an AdaBoost-based NIDS that uses statistical network flow features to detect malicious activities and respond in real time. This

allows for the possible analysis of HTTP, MQTT, and DNS network packets, reducing to a set of features from network flow [10].

Bhat et al. conducted a comprehensive comparative analysis of IDS and the Decision Tree (DT) binary classifier [7]. Their study aimed at introducing an innovative reduction feature strategy based on a swarm intelligence algorithm, Pigeon Inspired Optimizer (PIO), motivated by the navigational behavior of homing pigeons. The birds, while on their day-long journeys, engage in different kinds of maritime checks and sometimes change places with the other bird so that both stay in synchronization with the map and compass operator. This is the reason PIO was developed. The performance comparison was carried out through the UNSW-NB15, NSL-KDD, and KDDCup99 datasets of both Sigmoid PIO and Cosine PIO. Meanwhile, Sigmoid PIO selected ten features from KDDCup99, fourteen from UNSW-NB, and eighteen from NSL-KDD with respective accuracies of 94.7%, 86.1%, and 91.3%. The same line of work in [11] proposed utilizing five feature aggregations, including seven, five, and three flows from KDDCup99, UNSW-NB, and NSLKDD respectively, yielding 96, 91.7, 88.3% respectively for Cosine PIO (as well as other measurements). This complete comparison shows both the advantages and disadvantages of the feature reduction technique.

Khan et al. [13] conducted comprehensive experiments on four machine learning classifiers to investigate possible augmentations for improving their prediction performance on complex datasets: Random Forest, Extra Gradient Tree Boosting (XGBoost), Decision Tree, and Bagging Meta Estimator, in addition to one non-machine learning classifier, K-Nearest Neighbors (KNN). These studies sought to reduce the computing time while attempting to increase the accuracy of the models by selecting an 11-feature subset through the importance of the variables obtained by the Random Forest model. For the UNSW-NB dataset, accuracies were 74.87% for Random Forest, 71.43% for XGBoost, 74.64% for Bagging Meta Estimator, 74.22% for Decision Tree, and 71.10% for KNN. While XGBoost took more time to train, the forecasting time was a minimum for the Decision Tree technique and highest for KNN. Recent works highlighted such accuracy-computational efficiency trade-offs across various machine learning models in performance optimization [13].

Recently, Tama and Rhee [14] proposed a novel anomaly detection methodology using a gradient-boosting machine on datasets like NSL-KDD, UNSW-NB15, and GPRS. For this study, they used the other models: SVM, RF, CART, and DNN for comparison with GBM. Compared with the different models, GBM achieved higher accuracies: 91.31% on UNSW-NB, 91.82% on KDDTest+, and 86.51% on KDD-Test-21. Higher than all on each of the datasets in the statistics, the proposed implementation by the Python programming language and popular machine learning packages assured reliable comparisons with existing experiments. The results are evidence that GBM performs very well in the detection of anomalies and has enormous potential over traditional models.

V et al. developed a cloud-based intrusion detection system assisted by a Support Vector Machine and Artificial Neural Network. The attributes were selected using Univariate and Principal Component Analysis; the performance metrics are F1, precision, recall, and accuracy. The ANN and SVM approaches both captured the anomalies with a precision of 91% and 92%, respectively, given enough data. The research also emphasized the optimization and transformation of data engineering for improved accuracy. This research is vital in showing the advantages automation brings

in cloud-native intrusion detection and the importance of feature selection towards model reliability.

Kasongo and Sun [16] found five supervised models: SVM, LR, kNN, DT, and ANN using filter-based feature selection. They reduced their feature vector from the original 42 down to 19 features and improved the DT model binary classification accuracy from 88.13% to 90.85% with respect to the UNSW-NB15 dataset by using XGBoost. It covers binary and multiclass categorization, where the effect of significant feature reduction on model performances is shown. Results provide firm evidence to use XGBoost for optimization in feature selection and improvement in accuracy using different ML models.

Injadat et al. [18] proposed an oversampling-based NIDS framework with the help of ML techniques to detect attacks over the CICIDS 2017 and UNSW-NB 2015 data sets. The best feature selection was done by applying correlation-based feature selection (CBFS) and information gain-based feature selection (IGBFS), along with oversampling implemented using SMOTE. Compared to our work, in which we solved the whole dataset of UNSW-NB15 and optimized the training sample size and feature set, 99% accuracy is obtained for both datasets. Differentiation made in this work is that it considered the complete benchmark datasets and addressed the bias issues through the oversampling techniques. The results demonstrate the capabilities of feature selection and oversampling in enhancing the performance of intrusion detection systems.[19].

Henry et al. [20] achieved a promising accuracy of 89.4% using the ANN-based solution for network intrusion detection when implemented over the datasets of KDD9 and UNSW-NB15. This process uses a neural network to classify training data and to find optimal weights for the network's result. This architecture leverages all the neural network capabilities to improve the classifier's performance.

Moustafa et al. [21] utilized a beta mixture model for outlier detection and further fine-tuned the UNSW-NB15 dataset. ANNs were applied by Rehman et al. [22] using the intrusion detection of cryptography in IoT devices, giving an 84% accuracy with 8% false positive. These studies illustrate that the procedures based on ANN can be applied and become promising for improving network intrusion detection.

Jeong et al. [23] further went ahead to propose a two-stage hybrid model for anomaly detection, which involves both a well-thought-out structure of a CAE universal concept and a DNN deep neural network. Their model for the UNSW-NB15 dataset has given a test score of up to 91.29% in terms of accuracy. The proposed architecture is a good reflection of the complementary features existing in DNNs and CAEs in terms of feature engineering and increasing anomaly detection problems. This research points to the possibility of using different machine-learning techniques to enhance accuracy and reliability in existing intrusion detection systems. These results will help improve the response and performance of intrusion detection systems.

Moustafa et al. [24] explored multiple ML models. In this study, logistic regression and KNN give a straightforward and effective means of classification of information packets for the attributes and their analogs with other packets. The ensemble learning method consists of decision trees, extra trees, and random forests that can efficiently work with datasets of high dimensions, as they are interpretable and can be used for conducting analyses and classifying network packets. The boosted classifier operates such that an iterative aggregation of weak learners allows classification precision using

complex relationships within the dataset. The neural network models—MLP, MLP (Keras), GRU (Keras), and LSTM (Keras)—can learn complex patterns and dependencies within raw network packets while considering their sequential nature and intrinsic characteristics. Classification and comparison methodologies applied to the UNSW-NB 15 dataset can provide significant insights into themes related to network traffic patterns, differentiation of various packet types, and an assessment of the effectiveness of diverse machine-learning techniques for analyzing network packets.

Table 1 provides a summary of existing research on machine learning algorithms and feature selections on various datasets for anomaly detection by the accuracies and findings. The techniques, including some combinations of GBM, SVM, and random forests with optimization methods or feature reduction techniques, performed extremely well again on most of the datasets, and the robustness of these models can be considered.

Table 1. Machine Learning Showdown: Performance Highlights in Intrusion Detection

Study	Dataset	Machine Learning Algorithms	Feature Selection Method	Optimization Method	Accuracy (%)	Results Summary
Bhat et al. [7]	UNSW-NB15, NSL-KDD, KDDCup99	Decision Tree (DT)	Pigeon Inspired Optimizer (PIO)	N/A	94.7 (KDDCup99), 86.1 (NSL-KDD), 91.3 (UNSW-NB15)	Sigmoid PIO and Cosine PIO compared; Sigmoid PIO showed higher accuracies across datasets.
Khan et al. [13]	UNSW-NB	Random Forest, XGBoost, Decision Tree, Bagging Meta Estimator, KNN	Feature Significance (Random Forest)	N/A	74.87 (RF), 71.43 (XGBoost), 74.64 (Bagging), 74.22 (DT), 71.10 (KNN)	Random Forest and Bagging Meta Estimator performed best in terms of accuracy with moderate time.
Tama and Rhee [14]	UNSW-NB15, NSL-KDD, GPRS	GBM, SVM, Random Forest, CART, DNN	N/A	N/A	91.31 (UNSW-NB15), 91.82 (KDDTest+), 86.51 (KDDTest-21)	GBM outperformed other models in anomaly detection across multiple datasets.
V et al. [15]	UNSW-NB15	SVM, ANN	Univariate and PCA	N/A	91 (ANN), 92 (SVM)	SVM and ANN showed high accuracy; PCA effectively reduced features.
Kasongo and Sun [16]	UNSW-NB15	SVM, Logistic Regression, kNN, Decision Tree, ANN	Filter-based (XGBoost)	N/A	88.13 -> 90.85 (DT)	Feature reduction using XGBoost improved accuracy, especially for Decision Tree models.
Injadat et al. [18]	CICIDS 2017,	KNN, Random Forest	Correlation-based (CBFS),	SMOTE	99	Full dataset used; SMOTE and feature

	UNSW-NB 2015		Information Gain (IGBFS)			selection significantly improved accuracy.
Henry et al. [20]	KDD9, UNSW-NB15	ANN	N/A	N/A	89.4	ANN effectively categorized training data; beta mixture model refined dataset.
Moustafa et al. [21]	UNSW-NB15	N/A	N/A	Beta Mixture Model	N/A	Beta mixture model used for outlier detection; refined the UNSW-NB15 dataset.
Rehman et al. [22]	IoT Devices	ANN	N/A	N/A	84	ANN applied for cryptography intrusion detection; achieved moderate accuracy with low false positives.
Jeong et al. [23]	UNSW-NB15	CAE, DNN	N/A	N/A	91.29	Hybrid model combining CAE and DNN improved anomaly detection accuracy.

3 Methods and Materials

3.1 Dataset

Neither the historical trajectory of network traffic nor modern attack scenarios are well represented in the existing benchmark datasets. The UNSW Cyber Security Laboratory has produced a simulated environment to execute the UNSWNB15 test network. IXIA's new primary tool has allowed them to generate synthetic regular or pathological network traffic more realistically. The IXIA Perfect Storm application works well with UNSW-NB15 and enables it to mimic nine significant assault families. A total of 49 features were generated by examining the network packets using 12 algorithms and the Argus and Bro-IDS programs. However, prior benchmark datasets like KDD98, KDDCUP99, and NSLKDD focused on a limited number of attacks and included descriptions of outdated packets. Comparing the UNSW-NB15 dataset to the KDDCUP99 data set, considering factors such as normality, resource, quantity, and variety, demonstrates its advantages. It is believed that the UNSW-NB15 dataset will soon be an asset for the NIDS research community and

a state-of-the-art NIDS benchmark dataset [25, 26, 27]. Table 2 below gives a summary of the relevant points, helping us understand the background of the UNSW-NB15 dataset.

Table 2. Comparison of Network Traffic and Attack Scenario Datasets

Feature	UNSW-NB15	KDD98	KDDCUP99	NSLKDD
Year Created	2015	1998	1999	2005
Data Type	Network Traffic	Direct Marketing Campaign	Network Traffic	Network Traffic
Purpose	Network Intrusion Detection	Customer Response Prediction	Network Intrusion Detection	Network Intrusion Detection
Number of Records	2.5 million	95,412	4,898,431	125,973
Number of Features	49	479	41	41
Attack Types	9 attack categories	N/A	22 attack types	5 attack categories
Normal vs Attack Ratio	Balanced	N/A	Highly imbalanced	More balanced than KDDCUP99
Feature Types	Numerical, Categorical	Numerical, Categorical	Numerical, Categorical	Numerical, Categorical
Label Availability	Labeled	Labeled	Labeled	Labeled
Common Use Cases	Network security research, ML models	Marketing analytics, ML models	Network security research, ML models	Network security research, ML models
Data Source	IXIA PerfectStorm tool	Direct marketing data	Simulated network environment	Simulated network environment
Preprocessing	Minimal	Some preprocessing required	Some preprocessing required	Some preprocessing required
Publicly Available	Yes	Yes	Yes	Yes

3.2 Data pre-processing

One-hot encoding is a method employed to convert categorical variables into a form that can be provided to machine learning algorithms to perform better predictions. This technique transforms each possible value of a categorical feature into a binary vector, where only one bit is set to 1, representing the presence of that specific value, while all other bits are set to 0. Equ.1 shows the process:

$$One - hot(xi) = [0,0,\dots,1,\dots,0,0]One - hot(xi) = [0,0,\dots,1,\dots,0,0] \quad (1)$$

Here, xi means the ith category. When the vector value is 1, it indicates the position of the corresponding category of xi. to apply one-hot encoding, data must be encoded as a categorical integer matrix. This requires that the attribute values be first converted into integer labels. The label encoder is used to do it. It maps each category to a unique integer. The Equ.2 for label encoding can be expressed as:

$$LabelEncoder(x) = yLabelEncoder(x) = y \quad (2)$$

In Equ.2 x is the original categorical feature, and y is the corresponding integer label. This pre-processing step ensures the transformer works properly with the one-hot encoding standard. This process produces a sparse matrix. It has a dedicated column for each non-similar category. That is why, if an attribute has nn distinct values, then the corresponding one-hot encoded matrix will have nn columns. The transformation process is visualized as follows:

- Label Encoding:

Given a categorical feature $X = A, B, C$ the label encoder might produce:

$$A \to 0, B \to 1, C \to 2A \to 0, B \to 1, C \to 2$$

- One-Hot Encoding:

Transform the label-encoded values into one-hot encoded vectors:

$$0 \to [1,0,0],1 \to [0,1,0],2 \to [0,0,1]$$

This approach ensures that machine learning models can effectively interpret and utilize categorical data. That means it improves predictive accuracy and robustness in handling various data types [28].

3.3 Feature Selection

Calculating the relevance of a feature for any black-box estimator, classifier, or regressor involves quantifying predictive values under various permutations. The algorithm estimates the increase in prediction inaccuracy when a feature is missing. The error in prediction can be quantified using any chosen scoring criteria, such as the R^2 value in a regression model or classification accuracy measures [29]. Instead of removing features and retraining the estimator for each feature, the method introduces noise to the feature.

A comparison is made between the prediction error of the original dataset and the new dataset. If accurately predicting the target variable relies heavily on the column being randomly reshuffled, the model's predictions will be less accurate due to the random reordering. If the feature is not used by the model, there will be no change in prediction error.

-To calculate feature permutation and importance justifications, follow these primary steps:

1. Start with a previously developed machine learning model.
2. Determine the normal prediction error based on the data. Use a "test" dataset and a "train" dataset for each feature.
3. Change the dataset by switching the order of the feature columns.

4. Estimate the prediction error using the new dataset.

Note the relevance of the feature based on the difference between its baseline score and its score after the dataset has been shuffled.

For instance, examining the scrambled score (the baseline) illustrates this. It is recommended to repeat stages one to three several times and average the results to improve accuracy. The features are ordered by their contribution to the overall score of the model. Shuffling the features gives more importance to those affecting the score significantly. If randomly permuting a feature does not affect the forecast, it can be concluded that the feature's contribution to the model's predictions is equivalent to noise. In ADS feature permutation importance visualizations, negative feature significance values are capped at zero. Algorithm 1 illustrates feature permutation importance.

Algorithm (1): Algorithm Caption: Wolf Hunting Algorithm for Feature Selection

Let (N) *be the number of wolves in the population.*
Let (X_i) *represent the position of the* (i^{th}) *wolf in the feature space.*
Let $(F(X_i))$ *represent the fitness of the* (i^{th}) *wolf.*
Let (α, β, δ) *represent the positions of the top three wolves with the highest fitness.*
Let (C_i)*represent the hunting coefficient for the* $(i^{th})wolf$
1. Initialize wolf population:

$$[X_i^{(0)} \sim U(min, max) \quad for \quad i = 1,2, \dots, N]$$

2. Evaluate the fitness of each wolf:

$$[F(X_i) \quad for \quad i = 1,2, \dots, N]$$

3. Set the three wolves with the highest fitness as alpha, beta, and delta:

$$[\alpha, \beta, \delta = TopThree(X_i)]$$

4. For each iteration:
a. Calculate hunting coefficients:

$$[C_i = \frac{fitness(\delta) - F(X_i)}{fitness(\delta)} \quad for \quad i = 1,2, \dots, N]$$

b. Update the position of each wolf towards the three leading wolves based on the coefficients:

$$[X_i^{(t+1)} = X_i^{(t)} + C_i \cdot \left(\alpha - X_i^{(t)}\right) \quad for \quad i = 1,2, \dots, N]$$

c. Evaluate the fitness of each wolf:

$$[F\left(X_i^{(t+1)}\right) \quad for \quad i = 1,2, \dots, N]$$

d. Update alpha, beta, and delta if necessary:

$$[\alpha, \beta, \delta = TopThree\left(X_i^{(t+1)}\right)]$$

5. Return the features selected by the alpha wolf as the optimal feature subset:

$$[Optimal\ feature\ subset = \alpha]$$

3.4 Machine Learning Classifiers

- Logistic Classification

Logistic regression is a machine learning algorithm, and it is also a type of supervised learning. Typically, we use it in binary classification where our intent is to differentiate between one category of input (1) or another (0). The logistic function, also called the sigmoid function, is used for building a binary output variable with a relationship to the input features from a modelling perspective. The logistic function σ(z) is defined mathematically in equation (3)as follows:

$$[\sigma(z) = \frac{1}{1+e^{-z}}] \quad (3)$$

where (z) is a linear combination of the input features, expressed as in (4)

$$(z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n) \quad (4)$$

Here, $(\beta_0, \beta_1, \ldots, \beta_n)$ are the coefficients to be learned by the model, and $(x_1, x_2, \ldots, x_n)$ are the input features.

The logistic function maps any real-valued number to the interval (0, 1). This represents the likelihood of the output belonging to a particular class. The input is then classified into one of the binary classes by applying a threshold (usually 0.5). Logistic regression has been demonstrated to be used in network security. Using a logistic classifier a study classified network packets obtained in an unprocessed form. This study used the UNSW-NB 15 dataset [7]. It tried specifying the different kinds of network traffic [8]. The UNSW-NB 15 dataset has 49 features. There are 185,537 records of benign and attack instances. Logistic regression is excellent for binary classification problems and is one of the most popular machine learning algorithms data scientists use. As far as network security is concerned, logistic regression has been proven to be helpful in detecting suspected patterns and analyzing traffic, hence representing one more strength toward practical application. As Sun et al. reported, "logistic regression allows versatile security capabilities by giving interpretable insights and contributing to advancements for improved security solutions by modelling the relationship between input features and binary outcomes with the logistic function" [30, 31].

KNN (k-Nearest Neighbors)

The k-nearest neighbor algorithm is one of the most straightforward, easily understood, and versatile data-mining processes. It partitions data points based on their distance from other points in feature space. The method is referred to as k-neighbors because when a classification on a new data point is being made, the algorithm selects k nearest neighbors in the training set to ascertain which category it should assign. K-nearest neighbor is one widely accepted classifier based on the neighbor approach for data classification and clustering in research. For instance, Roeslin et al [32]. Conducted packet clustering in raw network packets to categorize different types of packets with similar packets using the KNN algorithm [32]. This similarity-based logic makes KNN useful when proximity between data points is considered essential during prediction.

Decision tree is a model in the form of a tree in which the data under consideration are recursively divided into two parts: the feature values *(x)* and the values associated with the feature *(y)*. Formally, in each split, a feature *xi* and a threshold *t* are selected, dividing the data as mentioned in Equ.5

$$(x, y)xi \leq t \mid (x, y) \mid xi \leq t \; and \; x, yxi > t \mid x, y \mid xi > t \quad (5)$$

The internal nodes represent the features that have been considered to make the decision, while the leaves are the final decision points based on these features, indicating the class labels at the last leaf nodes. In mathematical terms, the decision function for any non-leaf node is described as: If $xi \leq tx\_i \leq txi \leq t$ then go to the left child, else go to the right child. As described by Sun et al. [31], decision trees are a kind of model that represents a very high level of interpretability; this makes them applicable in packet analysis and comparison. Attributes such as simplicity and robustness are important to cope with various analytical problems in machine learning and data analysis.

- Extra Trees

An ensemble method that is also referred to as Extremely Randomized Trees is called the Extra Trees method, which is different from the ensemble methods we discussed. It is based on Decision Tree Learning, developing a network of trees and using trees for producing a final classification tree. It is especially good at dealing with high-dimensional feature space.

In mathematics, the whole core idea of the Extra Trees method can be written as follows: Given a dataset as presented in (6)

$$D = (xi, yi)i = 1nD = (xi, yi)_{i=1}^{n}D = (xi, yi)i = 1n \quad (6)$$

where xi are the input vectors, and yi are the target values, the algorithm builds MMM decision trees. For each decision tree ti, the construction process $t = 1$ to M begins. Every tree Tm is formed by selecting splits at random from the input features and then aggregating the predictions. The final prediction for a new input $x, y^{\hat{y}}y,$ is the average of the predictions from all trees as (7)

$$y = 1M \sum m = 1MTm(x)\hat{y} = \frac{1}{M}\sum_{m=1}^{M} T_m(x)y = M1 \sum m = 1MTm(x) \quad (7)$$

PCA is often used in combination with Extra Trees to improve its efficiency by reducing the dimensionality of the data. PCA converts the data to a lower dimension, including feature selection and dimensionality reduction tasks. Mathematically, PCA tries to find the projection of data into orthogonal axes and maximize the variance as in equation (8)

$$Z = XW \quad (8)$$

where X is the original data matrix, W is the matrix of eigenvectors of the covariance matrix of X, and Z is the transformed data in the reduced dimensional space. According to the study of [33, 52] the Extra Trees method is effective on raw network data. However, the Random Forest is an ensemble learning method. It means the aggregation of several decision trees to produce a prediction. It includes the making of several decision trees and their aggregation. Every tree constructed is represented by a random subset of features from the total set and by a random value of said subset. That is, a part of the training data is selected at random. The Random Forest algorithm is considered one of the most successful algorithms for solving overfitting problems. Ability to adjust to variations, the expertise of managing massive datasets, and the capability of dealing with information having different formats. Ji [9] suggests that it is possible to classify network packets and compare them in research works, through their use.

- Mathematical Representation:

1. Let Ti be the $i - th$ decision tree, where $i = 1,2, \ldots n.$

2. For each tree Ti , a random subset $Fi \subseteq FF\_i \subseteq FFi \subseteq F$ (where F is the total set of features) is selected.

3. The final prediction yˆ is calculated by aggregating the predictions $\widehat{y_i}$ of all n trees. It is expressed in (9)

$$y^{=}\frac{1}{n}\sum_{i=1}^{n}\widehat{y_i} \tag{9}$$

4. Let express the training dataset by D. For each tree Ti, a subset $Di \subseteq DD_i \subseteq DDi \subseteq D$ is randomly selected.

5. RF reduces overfitting by averaging multiple trees trained on different parts of the dataset. It follows equation (10)

$$Var(\hat{y}) = \frac{1}{n^2}\sum_{i=1}^{n}\mathrm{Var}(\widehat{y_i}) \tag{10}$$

Where $Var(\hat{y})$ is the variance of the aggregated prediction. This mathematical structure shows the robustness and adaptability of the RF algorithm. Ji [9] presented its applicability in classifying and comparing network packets.

- Gradient Boosting Classifier

Gradient Boosting is an ensemble learning technique. It combines multiple weak learners. These weak learners often represented as decision trees. This methodology involves an iterative model-building process. In this process misclassified instances are increasingly emphasized in subsequent models. The weak learners are defined as $h_1(x), h_2(x) \ldots\ldots h_m(x)$. Here each hi(x) is a decision tree model. The final model $F(x)$ is defined in equation (11)

$$F(x) = \sum_{i=1}^{m}\alpha_i h_i(x) \tag{11}$$

In this expression, α_i are the weights determined during the boosting process. The Gradient Boosting algorithm enhances the predictive performance by optimizing the loss function $L\big(y, F(x)\big)$. In this function, y represents the true value. At each stage t, the model $F_t(x)$ is updated as seen in (12)

$$F_t(x) = F_{t-1}(x) + \eta \cdot h_t(x) \tag{12}$$

In this equation η is the learning rate. The $h_t(x)$ represents the weak learner that fits the residual errors of the previous model $F_{t-1}(x)$This objective necessitates an accurate modelling of these relationships to yield reliable results. According to the categorization strategy suggested by Demir and Sahin [34], classes can be effectively separated, which enhances the model's ability to distinguish between different categories. Furthermore, the analysis and comparison of network packets require a detailed examination of their characteristics, which can be facilitated by the Gradient Boosting method. By analyzing the packets, one can identify patterns and anomalies, leading to better network performance and security.

- Neural Network MLP

The MLP stands for Multi-Layer Perceptron; it is a type of artificial neural net that is considered an improvement over the original type of neural net. MLP stands for Multi-Layer Perceptron, a network of linked nodes. In other terms, it can be said to be neurons. Many neurons have been organized in several layers. These models can easily and rapidly

be learned through knowledge. Boundary of the information along with its complex patterns and relationships. On the contrary, Han et al. [35] demonstrate. The learning process in MLPs is done through back-propagation and can even use a neural network. They are utilized as part of a network packet classification process via the differentiation of packet characteristics.

- Multi-Layer Perceptron MLP (Keras)

Keras is a high-level deep learning framework that, through an intuitive interface, allows one to get started. Keras is also one of the tools applied in neurocomputing to realize the Multi-Layer Perceptron (MLP), thus representing the essential and fundamental framework allowing a designer to define the network structure and activation. The main stages include function specification, model training, and so forth. The approach indicated by Alromaihi et al. [36] presents the application of a multilayer perceptron (MLP) as an alternative which is to be modelled. In this context, Keras will be employed for analysing and comparing the raw network packets under research investigation. Now to elaborate mathematically, the structure of an MLP can be defined in (13):

$$y = f(Wx + b) \tag{13}$$

where: y is the output, x represents the input features, W denotes the weight matrix, b stands for the bias vector, and f is the activation function applied to each neuron.by the way, the process of function specification involves defining this mathematical structure, while model training involves optimizing the parameters W and b using a training dataset to minimize a loss function L as seen in (14)

$$L = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, \hat{y}_i) \tag{14}$$

Where, n is the number of training examples, y_i are the true labels, $\hat{y}_i$ are the predicted labels, and $\mathcal{L}$ is the chosen loss function, such as mean squared error or cross-entropy.

Furthermore, Gated Recurrent Units (GRUs) is a Recurrent Neural Network (RNN) variant. It has been developed to construct models that have emerged recently. Using deep learning techniques, these units are designed to identify long-term dependencies within sequential data. Network packets can be analyzed by organizing packet data and determining the temporal sequential nature of the transmitted data [37]. The time-sequential characteristics of the data are captured during this process, which enables the prognosis and forecasting of trends based on the historical track record of the packets.

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN). Long Short-Term Memory (LSTM) networks are designed very specifically to have the capability to deal with limitations involved in traditional Recurrent Neural Networks while modeling and efficiently capturing long-range dependencies. Their cognitive abilities enable them to effectively retain and apply knowledge for extended durations, rendering them highly adept at scrutinizing consecutive data sets such as those found in network packets.

Algorithm (2) Classification Process:

1. Split the dataset into training and test sets
2. For each model in the list of models:
   a. Train the model on the training set using the selected features
   b. Predict on the test set
   c. Calculate evaluation metrics (F1 score, Accuracy, Precision, Recall)
   d. Store the metrics for comparison
3. Compare the models based on the stored metrics and select the best performing model

3.5 Evaluation Metrics

F1 Score Micro Measure For the study, the F1 Score Micro Measure will be used to review, compare and evaluate the results of all the algorithms of machine-learning. To achieve this goal, we followed an alternative method, implementing a machine-learning model optimizing for precision and largely recall based on the F1 Score heuristic during all steps when creating it. Besides the precision and recall of the model, the False negative (FN) and False positive (FP) rate is another major parameter which needs to be taken seriously. The F1 score is an excellent performance metric when the two groups to compare have a clear definition. We have finally chosen to work directly with the target variable because it would allow us to keep track of the sum of true and false positives/negatives across all categories [39].

3.5.1 F1 Score

In the realm of classification problems, the F1 score is an extremely important metric, especially when the precision-recall balance has paramount importance. This is particularly handy in cases of imbalanced datasets where one class could be in thousands while others could be below a hundred. The F1 score is a single metric summarizing a model's performance, balancing both precision and recall (i.e., it is the harmonic mean between the two). The F1 score is the harmonic mean of precision and recall. Precision relates to the percentage of all the instances we labeled that we predicted true positive ($true\ positive\ /\ total\ predicted\ positive$). Recall relates to the percentage of all the instances that were true positive that we labeled as true positive ($true\ positive/\ total\ actual\ positive$) as seen in (15)

$$F1\ Score = \frac{2TP}{(2TP+FP+FNN)} \tag{15}$$

3.5.2 Accuracy

One of the simplest and most used metrics for evaluating a classification model in machine learning is accuracy. This determines how accurately the model has predicted the instances in the dataset is called Accuracy. Hang et al. [41] suggest that the accuracy rate of our projections can be utilized as an approximate gauge of the dependability of this model. The proposed methodology involves an initial step of gathering the count of precise true positive (TP) and true negative (TN) projections, followed by a subsequent step of dividing the cumulative count by the overall count of projections executed, encompassing both the accurate and inaccurate ones (FP, FN) as noted in equation (16) below:

$$Accuarcy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (16)$$

Within the bounds of this range, some normalization is used to compensate and obtain some degree of precision in [0, 1]. Those poles are a stand-in for the extreme ends of the spectrum in how close a prediction is to perfectly forecast or nearly completely missing. Our model would not return any false positives nor can false negatives assuming it give an exhaustive prediction for all examples. If both the numerator and the denominator is one, the precision between them is also one. Those poles are pointed at the clear extremes of a prognosis' precision, between entirely precise and entirely imprecise. If our model were to produce a comprehensive prediction, we would not receive any false positives nor any false negatives. The precision is also one when both the numerator and the denominator are equal to unity. If all of the prognostications made by our system would prove false in a consistent manner then, no matter how far from the value of zero we sent the two numbers to reach a positive number or zero, the gap between zero and a positive number in our case would be equal to zero. It is also the case that the sum total of all true positive and true negative results will at all times add up to zero. If this happens, we can infer that our system always makes bad predictions. Parameswarappa et al. [42] No matter how many precise predictions in aggregate are made by a system consisting of that consistently produces a precise prediction; the value shall always equal zero in the event of a system defect. Where the precision score falls below 0.5, one can exchange the labels to create a more accurate prediction. The exact realm of true accuracy as defined by technical terms is vague and lies somewhere between 0.5 and 1.

If the data has been modified in any way, then providing a reliable assessment of the data accuracy is impossible. If the number of positive and negative labels is hugely unbalanced, the accuracy metric can be very deceiving and make you deduce something false. However, this is trivial 95% even when built as a "dummy" model to predict 0 all the time. A priori the models can be classified in two different categories: robust and fragile ones, such that robust means high resilience of models. This is correct, as the only possible classes can be predicted are 0 or 1. Though, in this case, this model is not accurate (that is just a measurement but does not even have a low false negative rate), and so accuracy is very inappropriate as a model metric for this data. These were caused by the faulty outputs the model produced. Add caption only using our model's accuracy to evaluate our performance will leave us with an inferior and less trustworthy service for our stakeholders and customers [40].

3.5.3 Precision

Precision, recall, and specificity are frequently utilized by data scientists as tactics to address the limitations imposed by accuracy. Through the examination of the level of precision in forecasting, it is possible to calculate the proportion of optimistic predictions that are correct. To calculate the accuracy rate, one can multiply the total number of correct predictions by the proportion of accurate forecasts out of the total number of forecasts made.

Consequently, an indication of the precision of the outcomes can be obtained through the evaluation of true positives (TP) and false positives (FP) which can be expressed in (17)

$$Precision = \frac{TP}{(TP+FP)} \quad (17)$$

3.5.4 Recall

Like the concept of accuracy, recall pertains to the total count of correctly answered questions. To determine the percentage of positive samples, one can calculate the total number of positive samples and divide it by the same number. This approach demonstrates efficacy irrespective of the expectation of successful outcomes for positive samples, including true positives and false negatives.

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

3.6 Gray Wolf Optimization (GWO)

The GWO is a metaheuristic algorithm. It follows the social structure and hunting behaviors of gray wolves [43, 44]. Multiple research show that enhances machine learning algorithms' ability to parse and categorize raw network packets by fine-tuning model parameters [45, 46]. One of the effective feature of GWO is that it maintains a balance between exploration and exploitation. That is why it can identify optimal solutions while avoiding local optima [47,48]. Its simplicity and minimal mathematical requirements make GWO a practical choice for improving machine learning algorithms. In this paper, the GWO has been used to enhance the precision of identifying raw network packets and optimize machine learning model parameters.

3.7 Study model

The tools of modeling support the prediction and examination of possible damage from a seismic event to the buildings. The dataset was classified into various class headers for the mere sake of organizing data. Introduction and loading of the dataset are the first steps. Finally, you need to deliver any necessary preprocessing of your input data. So, this process has the third phase in which slicing or partitioning the dataset into smaller subsets. At the fourth stage of the procedure, the dataset is divided into two sets: the training set and the test set. This is the measure for feature subset selection that is employed in the fifth stage to select a feature subset that best contributes to representation as a whole. Symmetrically stable unpredictability would mean that the kind of information available to us would have less qualitative benefit for reducing the degree of uncertainty. The sixth phase involves the training of models using datasets related to machine learning. Then, calculate the F1 score using the accuracy, precision, and recall matrices. The theoretical basis of our methodology is illustrated in Figure 1.

With the above steps followed, you can check how good the trained model is at predicting accurate forecasts from new data. To summarize, for the time being, our focus is on developing a machine-learning algorithm that can be used to detect and classify different types of nefarious behaviors that may occur on network packets. This suggests some evidence of the coverage and precision power of the model being tested as it is superior at functioning various attacks. It is therefore a reasonable choice for a metric in this context. An F1-score, specifically. F1-score: a metric that combines the balance between recall and precision (with values in [0, 1]).

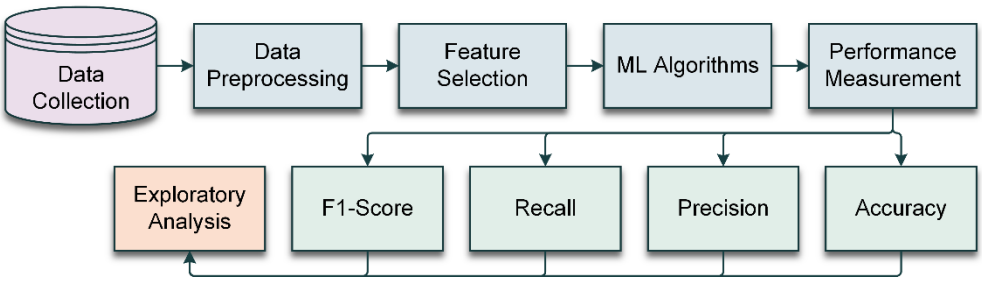

Figure 1. Methodology of our proposed system

4 Result and Discussion

This study used GWO to analyze a raw network packet dataset with various machine learning techniques. Among them, Forest (RF) and Extra Trees (ET) algorithms achieved the highest accuracy at 97.68% and 97.53%. Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) achieved accuracies of 96.48% and 96.33%. Multi-Layer Perceptron (MLP), Decision Tree, Gradient Boosting, k-Nearest Neighbors (kNN), and Logistic Regression showed satisfactory performance.

Feature selection is crucial in any machine learning algorithm with many features. This study included 39 initial features, such as 'srcip,' 'sport,' 'dstip,' and 'dsport.' Many feature configurations can bring out worthwhile insights, but one must be careful with the curse of dimensionality, which can lead to higher processing costs and overfitting machine learning algorithms. Feature selection is crucial as a less complex model reduces random noise overfitting. This approach can increase model accuracy by correcting for non-contributory factors affecting performance. Simplified feature sets also reduce training time and allow for easier searches for models and hyperparameters. Additionally, simpler models often generalize better to new, unseen data, and their interpretability is crucial when decisions must be understandable to the audience.

The feature selection of the Grey Wolf Optimization (GWO) in this work is significant. It uses a GWO approach inspired by the leadership hierarchy and hunting mechanism of gray wolves. Initially, potential solutions are symbolically introduced as “wolves” in the GWO. The top diplomat among these, the "alpha wolf," is followed by the "beta wolf" and the "delta wolf" in a command chain. Wolves regard the areas where the alpha pack members dwell as advantageous for improving their behavior. By using GWO, we reduced the dataset from 39 features to 19, indicating that some traits were less influential in categorization and needed less redundancy. This study shows that GWO has significantly improved feature selection. Reducing the number of features by 50% is more beneficial than over-parameterization by computationally expensive models like LSTM and GRU. Using a common feature subset across all models provides consistent evaluations, making it easier to compare results. Figure 2 presents the Feature Selection Process

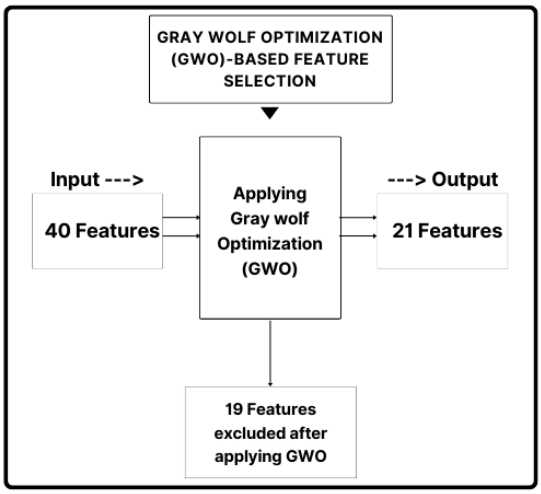

Figure 2. Feature Selection Process

Adding Grey Wolf Optimization significantly improves the performance of the algorithms, making better results in the classification of data using the parameters that come out of an optimization solution. The way in which the optimization method systematically explored the entire solution space was what enabled to improve the accuracy of several models. The implications of this study are important with respect to the detection and labeling of network intrusions. Accurate discovery of threats in networks is crucial in many applications, and the high precision of random forests and extra trees makes them potentially appealing. The use of LSTM and GRU models seems promising in terms of latent pattern detection in network traffic data. The feature selection results are shown in Figure 3. In addition, that study suggests that together with machine learning techniques, the results would enable developments in the domain of network security and motivate more investigation on raw network packets. The findings of this study could help security professionals and network administrators improve their intrusion detection systems and network security.

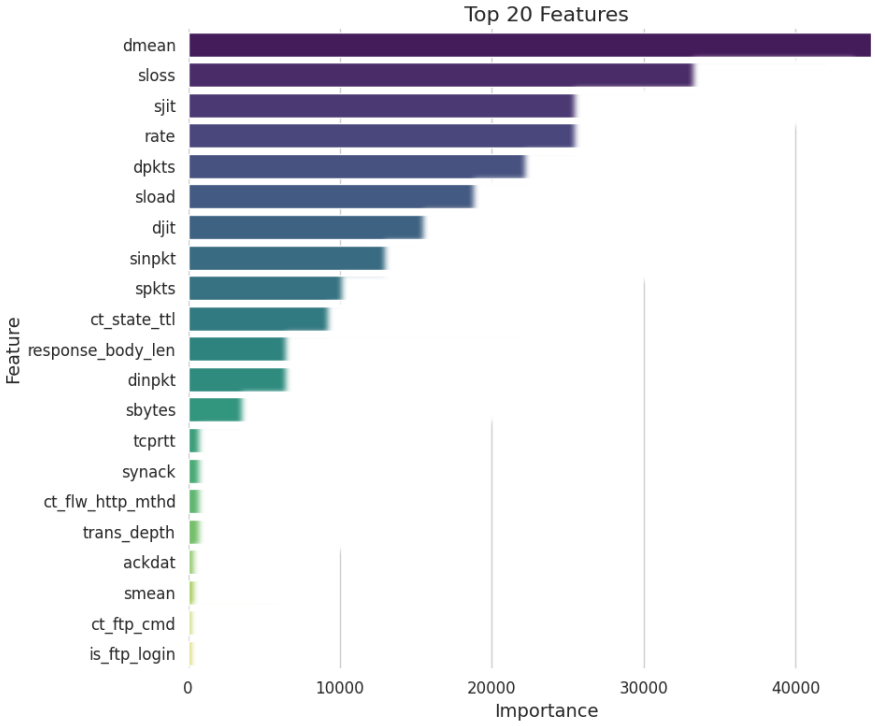

Figure 3. Feature Selection

The evaluation of classification models' performance is associated with a measure called accuracy. It is computed by dividing the number of correctly classified instances by the total number of instances. Its capacity to retrieve the true positive cases on the model is represented by the Sensibility, with the proportion of all the positive cases. Precision is a measure that refers to the accuracy of the positive instances that are predicted as such, and it is obtained by taking the true positives (TP) and dividing them by the total positive cases that are predicted. The F1-score is a metric that can give you a balance between precision and recall by computing the harmonic mean of the two. It can be handled by the F1-Score when there is a larger difference between the classes in society.

Table 3 shows the evaluation metrics results, an overall assessment and categorization of algorithms' performance. The talk is about the idea of unlikely woods. The best algorithm was the Random Forest algorithm and achieved 97.68 percent accuracy. Recall, precision, and F1-Score performance metrics have impressive values throughout. However, if we consider them with respect to other techniques, training and prediction times are generally good. Extra Trees performs with high precision with an accuracy of 97.53%, similar to Random Forest. It also shows a good F1 score with very good recall and specific precision. Both the training and prediction steps also take very little time.

The third ranked Keras LSTM (long short-term memory) model saw a precision rate of 96.48%. The metrics recall, precision, and F1-Score are found to be at great levels. However, this method takes much more time in the training process and also when the predictions have to be generated using this technique. The Keras built Gated Recurrent Unit (GRU) achieved 96.33%, comparable to Long Short-Term Memory (LSTM). It provides features based on recall, precision, and F1-Score. The processes of training and prediction consume a lot of time. The Multi-Layer Perceptron (MLP) using Keras gives an accuracy of 96.10%. The system routinely reaches good levels of accuracy, recall, and F1-Score. The prediction phase takes a bit longer than the training phase.

The Decision Tree algorithm performs better in metrics of accuracy, recall, precision, and F1-Score and has good prediction and training times. With the alternatives that we implemented in the predefined way, the best solution is the MLP (Multi-Layer Perceptron) model (without using Keras at all). The prediction time periods are much lower than in training. The Gradient Boosting Classifier has an accuracy score of 95.85%. Moreover, it performs well in recall, precision, and F1-Score. The training process of this method is very time-consuming, while there are a few other quick and easy-to-apply methods. The k-Nearest Neighbors (kNN) algorithm with a precision of 95.04% is shown. This model has a longer prediction time, but it performs well in recall, precision, and F1-score. The machine learning algorithm performances are evaluated using precision, recall, accuracy, and F1-Score. Key models used in this study include Random Forest, Extra Trees, Decision Tree, Gradient Boosting Classifier, k-Nearest Neighbors (kNN), and Logistic Regression. These models handle multi-dimensional data. This capability makes them suitable for packet analysis.

Several models, from Long Short-Term Memory (LSTM) to Generative Adversarial Networks (GANs) to Multi-Layer Perceptron (MLP), were developed to explore deep learning, some with and some without the Keras framework. Recurrent Neural Networks can clearly perceive sequential information in the input data. LSTM and GRU models are also helpful in Text Summarization. They measure key performance indicators that match their strengths and enable them to compete directly with high-end machine learning models. The ML model evaluation metrics used in this paper are accuracy, recall, precision, and F1-Score.

Table 3. Machine Learning Algorithms Evaluation Metrics.

Machine Learning Method	Accuracy	Recall	Precision	F1-Score
Logistic	92.803%	92.801%	92.832%	92.807%
K Nearest Neighbor	95.041%	95.041%	95.091%	95.057%
Decision Tree	96.382%	96.387%	96.383%	96.382%
Extra Trees	97.532%	97.534%	97.553%	97.531%
Gradient Boosting Classifier	95.854%	95.852%	95.868%	95.853%
Random Forest	97.687%	97.684%	97.698%	97.681%
Multilayer Perceptron	96.254%	96.251%	96.269%	96.252%
Multilayer Perceptron (Keras)	96.103%	96.105%	96.107%	96.106%
Gated Recurrent Unit (Keras)	96.331%	96.331%	96.339%	96.335%
Long Short-Term Memory (Keras)	96.483%	96.485%	96.488%	96.485%

The effectiveness of ML models is dependent on evaluation metrics. The investigation of the objective function at every iteration is a better approach to understanding the efficiency and convergence of the Grey Wolf Optimization algorithm. These values are observed during the iteration of the learning period. The experimental analysis shows that the objective function values dynamically change. When the values enter a constant state, it is considered the optimal learning rate value. In this study, it is 0.000412. In this study, the Grey Wolf Optimization algorithm has been used to find the model's parameters. That is why the precision, recall, accuracy, and F1-Score improve. The optimization process explained in this paper identifies parameters impacting the performance of ML models. The potential of these measurements to improve the accuracy of packet classification and analysis by optimizing machine-learning models is illustrated in Figure 3. Because of using the optimization algorithm and fine-tuning the parameters,the trained models offer a more effective defense against intrusion detection and classification.

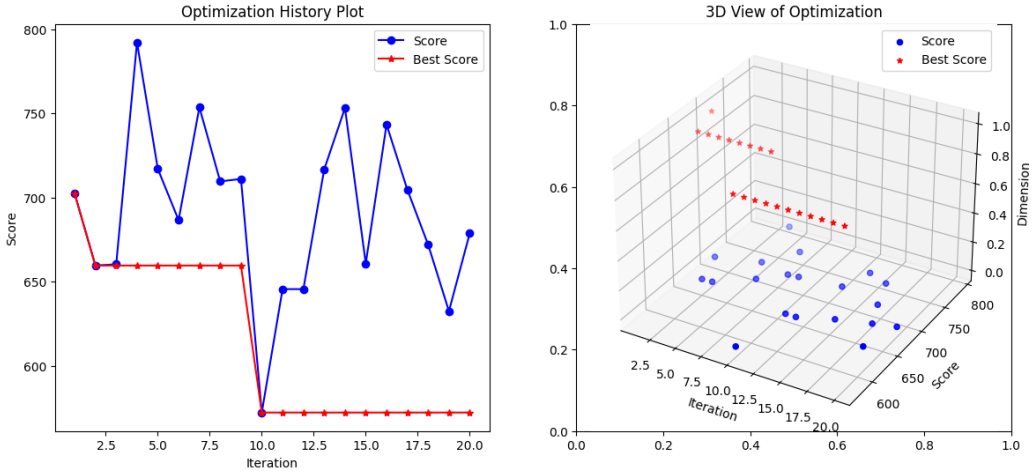

Figure.3. Gray wolf Optimization.

The investigation incorporated network packet data at its raw layer, with somewhat akin to GWO and other data machine learning classification methods. These were tested by accuracy, recall, precision, and F1-Score indicators. Among these, Random Forest and Extra Trees models achieved the highest accuracy rate of 97.68% and 97.53% respectively. Experiments with the ensemble-based methods illustrated that their performance turned out to be robust and worked well with raw network packet data, justifying the feasibility and effectiveness of them on dealing with high-dimensional datasets.

Gradient boosting classifiers, k-nearest neighbors, decision trees, and logistic regression are traditional machine learning methods that also performed well. Even though it might be less accurate, those are the ways our methodologies have helped us make some inference on the data. Deep learning models such as LSTM and GRU are good at processing sequential input data. The researchers then demonstrated that they were able to find temporal correlations in network traffic data to trace individual packets. Their accuracy on these tasks was 96.48% and 96.33% respectively.

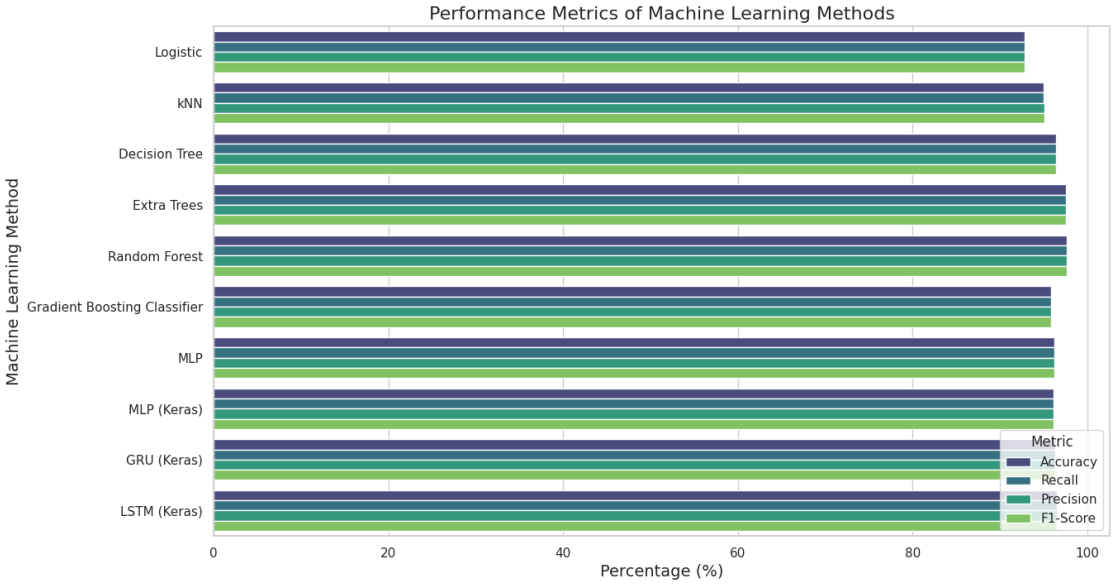

Figure 4. Performance of the Different ML Models

Figure 3 visually demonstrates the performance of the proposed experimental ML models in terms of different evaluation metrics. It shows that the Random Forest performs best for the experimental dataset, and the Extra Trees demonstrate similar performance. Although not as good as these two modes, Decision Tree's performance is indeed competitive. The logistic regression is the lowest-performing model. KNN performs better than logistic regression. However, it is not as good as tree-based approaches. The extended performance comparison illustrated in Figure 5 properly illustrates scenario.This is highly important in large datasets which contain large data hence machine learning relies mainly on feature selection for analyzing, using, and making essential decisions which are infeasible otherwise. The dataset had 39 features, e.g., 'srcip,' 'sport,' 'dstip,' and 'dsport'. A massive collection of features could lead to a better understanding of it, but at the same time, it brings its own set of challenges such as the curse of dimensionality. This entry may provide additional overhead to the computation and thereby generalize overfitting.

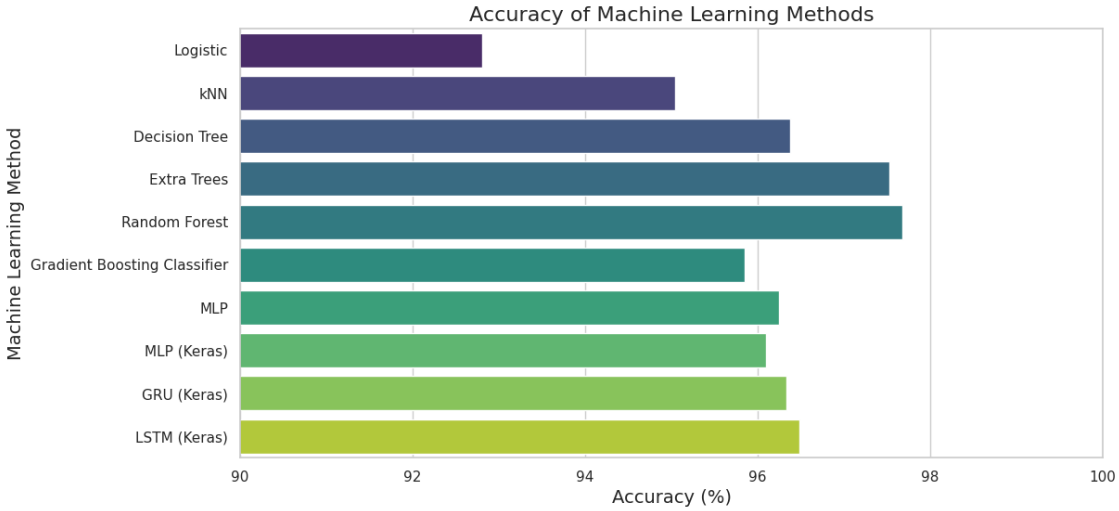

Figure 5. Extended performance comparison

So, the feature selection plays a very important part. If you are working with more than one model or hyperparameters, reducing the feature set can enhance the generalizability of the model for the new data, interpretability, and time taken in training.

GWO is actually critical when it comes to feature prioritization. Based on the social structure as well as hunting methods of gray wolves, the GWO algorithm optimized the feature space. The dimension of the dataset was a remained unresolved issue and GWO would extract the useful information from 39 attributes to 20 attributes, meaning most of the original attributes in the dataset were redundant or unimportant for classification. This had improved our algorithms remarkably, mainly due to the use of Grey Wolf Optimization (GWO). Using this optimization method boosts model accuracy significantly by simply trying all possible conditions.

Tracking objective function values in each iteration can achieve better presentation for the analysis of Grey Wolf Optimizer (GWO) convergence and the process of optimization. The algorithm has found the optimal solution when the values begin to constantly converge, which translates directly to a better, more accurate model! Upon stabilization, our mean value for the sample data would probably center itself around 0.000412. The fact that the algorithm converges demonstrates this by finding optimal solutions and changing parameters. This study unearthed loopholes in network security and their classification. Network Risk Identification is very important in almost all real-world situations. Results of the Random Forests, Extra Trees, and Neural Network models -- LSTM and GRU, respectively -- seemed promising, making them eligible for real-world deployments. Moreover, this study provides the foundation to begin using sophisticated machine learning techniques to process raw network packets. Security professionals, including network administrators, can make use of the pattern knowledge available from the dataset provided to tune up network security and make the intrusion detection systems behave more like real attackers.

5 Performance Comparison

The models studied in this paper have been compared with other similar approaches which has been listed in Table 4. All methods presented in Table 4 has been studied in the UNSW-NB15 dataset. The comparison shows that CSK-CNN and Bi-LSTM achieve accuracies of 99.60% and 99.52%, respectively. These are the highest accuracies.

Table 4. Performance comparison

Method	Machine Learning Method	Accuracy	Recall	Precision	F1-Score
Proposed	Logistic	92.803%	92.801%	92.832%	92.807%
	K Nearest Neighbor	95.041%	95.041%	95.091%	95.057%
	Decision Tree	96.382%	96.387%	96.383%	96.382%
	Extra Trees	97.532%	97.534%	97.553%	97.531%
	Gradient Boosting Classifier	95.854%	95.852%	95.868%	95.853%
	Random Forest	97.687%	97.684%	97.698%	97.681%
	Multilayer Perceptron	96.254%	96.251%	96.269%	96.252%
	Multilayer Perceptron (Keras)	96.103%	96.105%	96.107%	96.106%
	Gated Recurrent Unit (Keras)	96.331%	96.331%	96.339%	96.335%
	Long Short-Term Memory (Keras)	96.483%	96.485%	96.488%	96.485%
More et al. [49]	Random Forest	99.45%	99.55%	99.50%	99.53%
	Decision Tree	98.77%	96.28%	98.79%	97.53%
	Gradient Boosted Decision Tree	98.73%	97.50%	97.40%	97.45%
	Logistic Regression	98.93%	98.95%	98.92%	98.94%
Chavan et al. [50]	Linear SVM	99.16%	99.16%	99.16%	99.16%
	Bi-LSTM	99.52%	99.54%	99.50%	99.52%
CSK-CNN [51]	CSK-CNN	99.60%	99.62%	99.58%	99.60%

6 Discussion

The purpose of this comparison is to assess the extent to which our methods are poised to deliver comprehensive results and to draw attention to any potential insights missed without our method when measured against the conventional methods in the current state of the art. It is a benchmark not only to compare our results but also to have a holistic view of how our method fares against other existing methods. With the introduction of modern optimization algorithms such as Grey Wolf Optimization (GWO), the focus on feature selection deteriorated the traditional approaches, and a major jump took place. Some of the common techniques used before are Recursive Feature Elimination (RFE), Principal Component Analysis (PCA), and correlation matrices. Although such traditional methods have their advantages, they may lack the ability to effectively capture the complexities of intricate datasets, in particular those that are created based on raw network packets. Conversely, GWO can explore, for more extensive, the feature space of computations, potentially obtaining more optimal feature subsets as it employs a natural heuristic.

Particle Swarm Optimization (PSO), Genetic Algorithms (GA), and other optimization techniques are commonly used in similar applications. While ideal in most settings, GWO's

unique hierarchical structure and hunting approach based on the hunting strategy provide a skilled search mechanism which can outperform traditional algorithms with particular types of data in specific conditions. Deep learning models, especially with Recurrent Neural Networks like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), have obtained better performance and proven their capability to process sequences over the last years. However, these models are associated with the downsides of longer training time and requiring more data. Less complex traditional machine learning models deliver the same performance and are more cost-effective from a computational perspective when paired with optimized features. For the experiment we performed, we used ensemble methods like Random Forest and Extra Trees to join the outputs on different models for an increase in performance. There are additional advantages of AdaBoost implementation, whereas AdaBoost is an ensemble method of training based on another kind of ensemble method of training known as bagging. Trade-offs of these three aspects should be analyzed and compared, with respect to accuracy, bias-variance structure, and computation efficiency. Using GWO, we applied automated feature selection. Another option is expensive manual feature engineering either through domain knowledge-based feature crafting or transfer. That being said, handcrafting feature engineering is tedious work, and it will not catch the less explicit interactions among features that a computer-aided technique can savour. The proposed methodology integrated with GWO to compare with other methods to present a holistic view of research directions. This approach has been successful, but it is important to know the pros and cons of all methods. Table 5 illustrates the Computational Trade-Offs between the Deep Learning and Traditional Classifiers.

Table 5: Computational Trade-Offs between the LSTM, GRU, and Traditional Classifiers.

Feature	Deep Learning (LSTM, GRU)	Traditional Classifiers (SVM, RF, KNN, etc.)
Computational Cost	High which requires extensive training, backpropagation through time (BPTT) for LSTMs/GRUs.	Low to moderate that faster training, especially for linear models.
Memory Requirements	High requirement that requires storing weights, gradients, and activations over multiple time steps.	Low requirement by storing fewer parameters depending on the model.
Training Time	Slow training time due to sequential data processing and complex architectures.	Fast training time for most traditional classifiers converges quickly, especially with small datasets.
Inference Time	Slow to moderate Inference Time that Requires passing through multiple layers.	Fast for simple models (like logistic regression) can perform inference in milliseconds.
Interpretability	Low Interpretability for the Deep learning models are often black-box approaches.	High Interpretability for Decision trees, linear models, and rule-based classifiers offer better interpretability.
Data Requirement	High requirement as it needs large labeled datasets for effective learning.	Low to moderate requirement that can work well with smaller datasets and handcrafted features.
Scalability	Moderate to high Scalability range which scales well with hardware (GPUs, TPUs).	High scalability that handles large datasets efficiently with appropriate feature engineering.
Suitability for Sequential Data	High suitability for LSTMs and GRUs are specifically designed for sequential data like time series or NLP.	Low to moderate suitability because it needs feature engineering (e.g., lag features) for sequential data.

In addition to Table 6 that shows the weakness and strength comparison between GWO, PCA and RFE.

Table 6: Comparison given strength and weakness between GWO, PCA and RFE.

Feature Selection Method	Strengths	Weaknesses
Grey Wolf Optimizer (GWO)	Metaheuristic optimization which can find optimal feature subsets, suitable for high-dimensional data.	Computationally expensive which may not always find the globally optimal subset.
Principal Component Analysis (PCA)	Reduces dimensionality while preserving variance how it removes collinearity so it speeds up training.	Loses interpretability which does not select original features but transforms them.
Recursive Feature Elimination (RFE)	Iteratively removes least important features based on model performance that maintains interpretability.	Computationally expensive for large feature sets depending on the choice of the base estimator.

7 Conclusion and Future Works

The evaluation and comparison of original unprocessed network packet data, Techniques, and Grey Wolf Optimization (GWO). In this research, a structured assessment was carried out to determine the effectiveness of multiple machine learning algorithms like logistic regression, k-Nearest Neighbours, decision trees, extra trees, random forests, gradient boosting classifiers, multi-layer perceptron (MLP), MLP (Keras), GRU (Keras), and LSTM (Keras) in combination with the Grey Wolf Optimization approach. In the results, the Random Forest has the best metrics, which is why the Random Forest and Extra Trees can be considered some of the highest performing classifiers, with an accuracy of 97.68%. Now the algorithms achieve better accuracy for the same set of parameters, due to the parametric optimization of the models using the GWO integration. This optimization process to produce satisfactory solutions leading to significant classification improvements and model efficiency. This has significant implications for tasks such as network intrusion detection and classification. Random Forest, Extra Trees, LSTM, and GRU are all models with high accuracy that can be widely used, especially in network threat detection. By the use of these models, network administrators and security analysts are given resources which help them to strengthen the network defences, by these the risk of future security breaches also minimize. The testing for this project was conducted with several performance measures, which are accuracy, recall, F1-score, and precision. These methods made it possible to perform qualitative comparisons and gave a quantitative evaluation of the models. The selection of suitable algorithms and optimization methods under assessment standards of network packet analysis can guide researchers and users to choose models well. Moreover, it could further investigate whether these optimized performance solutions generalize to other networks to maintain their effectiveness. Machine learning algorithms and optimization techniques may encounter the various evolving problems of network intrusion detection and classification by learning newer data while improving their strategies.

Based on the results of this study, several research directions are suggested for future studies. In the next step, researchers can study the generalization and performance of these algorithms and optimization techniques for other network environments including a variety of topologies, protocols, traffic patterns, etc. This can enable the generalization of the findings and robustness across different settings. This will result in real-time Intrusion Detection systems that use both GWO and a set of trained complex machine learning models to immediately detect and respond to threats that have a very low false positive and

negative rate. This means the algorithms must be optimized for low latency and high throughput, to process real-time data streams efficiently. Hybrid models combining GWO with other optimization methods may lead to improved feature selection since GWO appears to be a good one. Furthermore, we need to look at automation when performing the feature engineering as well, ideally developing feature-engineering methods that can adapt to the nature of new data. By studying various ensemble learning techniques, for instance, stacking, blending, and hybrid ensembles, one would be able to learn how to accomplish enhanced accuracy and robustness in network intrusion detection. It is critical to know how robust they are against adversarial attacks. We will look at some promising directions, such as how to make models which are robust to adversarial attacks, so the intrusion detection systems are effective facing advanced attacks. There needs to be research on how to scale these algorithms to large-scale network settings. This involves improving the computational efficiency of the models and looking into distributed computing methods to work with massive network traffic datasets. Longitudinal studies that measure the performance of these models over time can help us understand how well the model measures up and changes with threats to the network. Slightly to the right you can use GWO with automated feature selection. Another option to consider is manual feature engineering, which consists of a number of crafted features, typically on domain-specific knowledge.

References

[1] Kenyon, L., Deka, L., & Elizondo, D. (2020). Are public intrusion datasets fit for purpose? Characterising the state of the art in intrusion event datasets. *Computers & Security*, (Vol 99, pp 102022). https://doi.org/10.1016/j.cose.2020.102022

[2] von Solms, R., & van Niekerk, J. (2013). From information security to cyber security. *Computers & Security*, (Vol.38, pp 97-102). https://doi.org/10.1016/j.cose.2013.04.004

[3] Ainslie, S., Thompson, D., Maynard, S., & Ahmad, A. (2023). Cyber-threat intelligence for security decision-making: A review and research agenda for practice. Computers & Security, (Vol.132, pp 103352). https://doi.org/10.1016/j.cose.2023.103352

[4] Jesus, V., Bains, B., & Chang, V. (2023). Sharing is caring: Hurdles and prospects of open, crowd-sourced cyber threat intelligence. IEEE Transactions on Engineering Management, (pp1-20). https://doi.org/10.1109/TEM.2023.3279274

[5] Zainel, H., & Koçak, C. (2022). LAN intrusion detection using convolutional neural networks. Applied Sciences, (Vol. 12(13), pp.6645). https://doi.org/10.3390/app12136645

[6] Mohamad Asri, R., & Khairuddin, I. E. (2022). A theoretical framework for the awareness of phishing attack. Journal of Information and Knowledge Management (vol. 1, pp 126-135).

[7] Bhat, P., Behal, S., & Dutta, K. (2023). A system call-based Android malware detection approach with homogeneous & heterogeneous ensemble machine learning. Computers & Security, (Vol. 130, pp.103277). https://doi.org/10.1016/j.cose.2023.103277

[8] Tessarolo, F., et al. (2021). Developing ambient assisted living technologies exploiting potential of user-centred co-creation and agile methodology: The Captain

Project experience. Journal of Ambient Intelligence and Humanized Computing. https://doi.org/10.1007/s12652-021-03649-0

[9] Lee, S., Nguyen, N., Karamanli, A., Lee, J., & Vo, T. P. (2022). Super learner machine-learning algorithms for compressive strength prediction of high-performance concrete. Structural Concrete, (Vol.24(2), pp.2208-2228). https://doi.org/10.1002/suco.202200424

[10] Sarhan, M., Layeghy, S., Moustafa, N., & Portmann, M. (2021). NetFlow datasets for machine learning-based network intrusion detection systems. In Big data technologies and applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10 (pp. 117-135). Springer International Publishing.

[11] Ji, M., Liu, L., Du, R., & Buchroithner, M. F. (2019). A comparative study of texture and convolutional neural network features for detecting collapsed buildings after earthquakes using pre-and post-event satellite imagery. *Remote Sensing,* (Vol 11(10), pp.1202).

[12] Disha, R. A., & Waheed, S. (2022). Performance analysis of machine learning models for intrusion detection system using Gini impurity-based weighted random forest (GIWRF) feature selection technique. *Cybersecurity*, (Vol. 5(1), pp.1)

[13] Khan, P. W., Byun, Y. C., & Jeong, O.-R. (2023). A stacking ensemble classifier-based machine learning model for classifying pollution sources on photovoltaic panels. *Scientific Reports,* (Vol.13(1)). https://doi.org/10.1038/s41598-023-35476-y

[14] Tama, A., & Rhee, K.-H. (2017). An in-depth experimental study of anomaly detection using gradient boosted machine. *Neural Computing and Applications,* (Vol.31(4), pp. 955–965). https://doi.org/10.1007/s00521-017-3128-z

[15] M. M. V., Mewada, B. R. R., & B. D. (2022). Hybrid machine learning approach-based intrusion detection in cloud: A metaheuristic assisted model. *Multiagent and Grid Systems,* (Vol.18(1), pp. 21-43). https://doi.org/10.3233/MGS-220360

[16] Kasongo, S. M., & Sun, Y. (2020). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. *Journal of Big Data,* (Vol.7(1), 1-20).

[17] Anushiya, R., & Lavanya, V. S. (2023). A new deep learning with swarm-based feature selection for intelligent intrusion detection for the Internet of Things. *Measurement: Sensors,* (Vol.26, pp.100700).

[18] Injadat, M., Moubayed, A., Nassif, A. B., & Shami, A. (2021). Multi-stage optimized machine-learning framework for network intrusion detection. *IEEE Transactions on Network and Service Management,* (Vol.18(2), pp.1803-1816). https://doi.org/10.1109/TNSM.2020.3014929

[19] Santhadevi, & Janet, B. (2023). Stacked deep learning framework for edge-based intelligent threat detection in IoT network. *The Journal of Supercomputing,* (pp.1-34).

[20] Henry, et al. (2023). Composition of hybrid deep learning model and feature optimization for intrusion detection system. *Sensors,* (Vol.23(2), pp.890). https://doi.org/10.3390/s23020890

[21] Moustafa, N., Creech, G., & Slay, J. (2017). Big data analytics for intrusion detection system: Statistical decision-making using finite Dirichlet mixture models. In *Data analytics and decision support for cybersecurity* (pp. 127-156). Springer.

[22] Rehman, et al. (2022). Intrusion detection based on machine learning in the Internet of Things, attacks, and countermeasures. *The Journal of Supercomputing,* (Vol.78(6), pp.8890-8924). https://doi.org/10.1007/s11227-021-04188-3

[23] Jeong, K.-J., et al. (2022). Two-stage deep anomaly detection with heterogeneous time series data. *IEEE Access,* (Vol.10, pp. 13704-13714). https://doi.org/10.1109/ACCESS.2022.3147188

[24] Koroniotis, N., Moustafa, N., & Slay, J. (2022). A new intelligent satellite deep learning network forensic framework for smart satellite networks. *Computers and Electrical Engineering,* (Vol. 99, pp.107745).

[25] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive dataset for network intrusion detection systems (UNSW-NB15 network dataset). In *2015 Military Communications and Information Systems Conference (MilCIS).*

[26] Moustafa, N., & Slay, J. (2016). The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 dataset and the comparison with the KDD99 dataset. *Information Security Journal: A Global Perspective,* (Vol. 25(1-3), pp.18-31).

[27] Moustafa, N., Slay, J., & Creech, G. (2017). Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. *IEEE Transactions on Big Data,* (Vol.5(4), pp. 481-494).

[28] Almarshdi, R., Nassef, L., Fadel, E., & Alowidi, N. (2023). Hybrid deep learning-based attack detection for imbalanced data classification. *Intelligent Automation & Soft Computing,* (Vol.35(1), pp.297-320).

[29] Shaikh, & Gupta, P. (2022). Real-time intrusion detection based on residual learning through ResNet algorithm. *International Journal of System Assurance Engineering and Management,* (pp.1-15).

[30] Hidayat, S. N., et al. (2022). Hybrid learning method based on feature clustering and scoring for enhanced COVID-19 breath analysis by an electronic nose. *Artificial Intelligence in Medicine,* (pp.102323).

[31] Sun, H., Burton, H. V., & Huang, H. (2021). Machine learning applications for building structural design and performance assessment: State-of-the-art review. *Journal of Building Engineering,* (Vol.33, pp.101816).

[32] Roeslin, S., et al. (2020). A machine learning damage prediction model for the 2017 Puebla-Morelos, Mexico, earthquake. *Earthquake Spectra,* (Vol.36(2), pp.314-339).

[33] Kabir, M. A. B., Hasan, A. S., & Billah, A. M. (2021). Failure mode identification of column base plate connection using data-driven machine learning techniques. *Engineering Structures,* (Vol. 240, pp. 112389).

[34] Demir, S., & Sahin, E. K. (2022). Comparison of tree-based machine learning algorithms for predicting liquefaction potential using canonical correlation forest, rotation forest, and random forest based on CPT data. *Soil Dynamics and Earthquake Engineering,* (Vol.154, pp.107130).

[35] Han, J., et al. (2020). Seismic vulnerability assessment and mapping of Gyeongju, South Korea using frequency ratio, decision tree, and random forest. *Sustainability*, (Vol.12(18), pp.7787).

[36] Alromaihi, N., & Al-Omary, A. Y. (2022). Machine learning and big data-based IDS system extensive survey. In *2022 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)* (pp. 7-12).

[37] Rajasekaran, P., & Magudeeswaran, V. (2023). Malicious attacks detection using GRU-BWFA classifier in pervasive computing. *Biomedical Signal Processing and Control*, (Vol.79, pp.104219).

[38] Velliangiri, S., et al. (2023). An enhanced security framework for IoT environment using Jaya optimization-based genetic algorithm. *International Journal of Internet Technology and Secured Transactions*, (Vol.13(1), pp.11-25).

[39] Arregoces, P., et al. (2022). Network-based intrusion detection: A one-class classification approach. In *NOMS 2022—2022 IEEE/IFIP Network Operations and Management Symposium* (pp. 1-6).

[40] Alqahtani, S. (2022). FSO-LSTM IDS: Hybrid optimized and ensembled deep-learning network-based intrusion detection system for smart networks. *The Journal of Supercomputing*, (Vol.78(7), pp. 9438-9455).

[41] Hang, et al. (2023). Network intrusion anomaly detection model based on multiclassifier fusion technology. Mobile Information Systems, 2023.

[42] Parameswarappa, P., Shah, T., & Lanke, G. R. (2023). A machine learning-based approach for anomaly detection for secure cloud computing environments. In 2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT) (pp. 931-940).

[43] Kaveh, & Servati, H. (2001). Design of double-layer grids using backpropagation neural networks. Computers & Structures, (Vol.79(17), pp.1561-1568).

[44] Kaveh, et al. (2008). Optimal design of transmission towers using genetic algorithm and neural networks. International Journal of Space Structures, (Vol.23(1), pp. 1-19).

[45] Kaveh. (2014). Advances in metaheuristic algorithms for optimal design of structures. Springer International Publishing, (pp.9-40).

[46] Kaveh, & Khalegi, A. (1998). Prediction of strength for concrete specimens using artificial neural networks. Advances in Engineering Computational Technology, (pp.165-171).

[47] Kaveh, & Khavaninzadeh, N. (2023). Efficient training of two ANNs using four meta-heuristic algorithms for predicting the FRP strength. Structures, (Vol.52, pp.256-272).

[48] Kaveh, & Rad, A. S. (2023). Metaheuristic-based optimal design of truss structures using algebraic force method. Structures, (Vol.50, pp.1951-1964).

[49] More, S., Idrissi, M., Mahmoud, H., & Asyhari, A. T. (2024). Enhanced intrusion detection systems performance with UNSW-NB15 data analysis. Algorithms, (Vol.17(2), pp. 64).

[50] Chavan, P., Hanumanthappa, H., Satish, E. G., Manoli, S., Supreeth, S., Rohith, S., & Ramaprasad, H. C. (2024). Enhanced hybrid intrusion detection system with attention mechanism using deep learning. SN Computer Science, (Vol.5(5), pp.534).

[51] Song, J., Wang, X., He, M., & Jin, L. (2023). CSK-CNN: Network intrusion detection model based on two-layer convolution neural network for handling imbalanced dataset. Information, (Vol. 14(2), pp.130).

[52] Abunasser, B. S., Daud, S. M., & Abu-Naser, S. S. (2023). Predicting stock prices using artificial intelligence: A comparative study of machine learning algorithms. International Journal of Advances in Soft Computing and Its Application, (Vol.15(3), pp.41-53). https://doi.org/10.15849/IJASCA.231130.03

Hussein Ahmad Al Ofeishat is an Associate Professor at Al Balqa Applied University, specializing in Computer Engineering. He earned his Ph.D. in Computer Engineering from the Faculty of Computer Engineering at the National Technical University of Ukraine in 2005, following his MSc. in Electrical Engineering from the same university's Faculty of Electrical Engineering in 1992. His research interests focus on Computer Networks and Network Security.

Jawdat S. Alkasassbeh was born in 1983 in Jordan. He obtained a bachelor's degree in Communications Engineering from the Department of Electrical Engineering, Faculty of Engineering, Mutah University, Al-Karak, Jordan, in 2006, and a master's degree in Communications Engineering from the University of Jordan, Amman, Jordan, in 2011. Alkasassbeh obtained his Ph.D. degree from the School of Mechanical Engineering and Electronic Information at China University of Geosciences in Wuhan, China, in 2021. His current research interests include applications of evolutionary algorithms, applied artificial intelligence (AI), power reduction in mobile communication mechanisms, digital wireless communication systems.

Albara Awajan is a professor in the Cyber Security Department, Faculty of Artificial Intelligence, at Al-Balqa Applied University, Jordan. He received his B.C.S. degree from Mutah University in 2001, followed by an M.C.S. in Multimedia and Internet Computing and a Ph.D. in Computer Networks and Multimedia Applications from the University of Glamorgan, U.K., in 2003 and 2008, respectively. Dr. Awajan's research interests include IoT security and intrusion detection, with a particular focus on enhancing the security and reliability of networked systems. His work explores advanced threat detection mechanisms, secure communication protocols, and the optimization of cybersecurity frameworks for emerging technologies.

Moutaz Alazab received the B.Eng. degree in computer engineering from the Faculty of Engineering, Al-Balqa Applied University, Jordan, in 2009, and the Ph.D. degree in cybersecurity from Deakin University, Australia, in 2014. During the Ph.D. degree, he was an Active Scholar with the Securing Cyberspace Laboratory and the Network and System Security Laboratory (NSCLab). He is currently a Computer Security Expert with industry, academic, and research experience.