

Enhanced Network Communication Security Through Hybrid Dragonfly-Bat Feature Selection for Intrusion Detection

Mosleh M. Abualhaj ^{1,*}, Sumaya N. Al-Khatib ², Mahran Al-Zyoud ¹, Iyas Qaddara ²,
Mohammad O. Hiari ¹, and Sultan Mesfer A. Aldossary ³

¹ Department of Networks and Cybersecurity, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan

² Department of Computer Science, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan

³ Department of Computer Engineering and Information, College of Engineering in Wadi Alddawasir, Prince Sattam University, Saudi Arabia

Email: m.abualhaj@ammanu.edu.jo (M.M.A.); sumayakh@ammanu.edu.jo (S.N.A-K.); m.zyoud@ammanu.edu.jo (M.A-Z.); i.qaddara@ammanu.edu.jo (I.Q.); m.hyari@ammanu.edu.jo (M.O.H.); s.aldossary@psau.edu.sa (S.M.A.A.)

*Corresponding author

Abstract—Network Intrusion Detection and Prevention Systems (NIDPS) play a critical role in securing network communications by detecting and mitigating cyber threats. Machine Learning (ML)-based NIDPS have proven to be highly effective in identifying network intrusions; however, their performance deteriorates when dealing with high-dimensional data. To address this, an efficient feature selection technique is essential to eliminate redundant or less relevant features, enhancing both accuracy and computational efficiency. A novel hybrid feature selection approach combining the Dragonfly Algorithm (DA) and Bat Algorithm (BA) is proposed to reduce dimensionality. Using the optimized feature subset, classification is performed with Extreme Gradient Boosting (XGBoost), Random Forest (RF), Logistic Regression (LR), and Support Vector Machine (SVM). This study employs the UNSW-NB15 dataset to train and evaluate an NIDPS framework. Experimental results demonstrate that the DAuBA feature selection method significantly improves classification performance, with XGBoost and Decision Tree (DT) achieving 100% accuracy, highlighting the effectiveness of the suggested approach in intrusion detection.

Keywords—Intrusion detection, machine learning, feature selection, dragonfly algorithm, and bat algorithm

I. INTRODUCTION

A huge number of communications is running over IP networks. The traffic of these communications is susceptible to several types of threats. Therefore, cybersecurity should offer advanced methods to secure the traffic of network communications [1, 2]. Two significant innovations have emerged in cybersecurity in recent years: Machine Learning (ML) and deep learning. Because these

technologies excel at finding patterns and solving problems when there are many possibilities, they are perfect for catching intrusions in computer networks. In fact, some experts think that in just a few years, ML will become the main method for Network Intrusion Detection and Prevention Systems (NIDPS) work [3, 4]. NIDPS is a tool that is used to secure the network's inbound and outbound communications. For ML to achieve optimal performance, attacks must occur in a manner that is sufficiently similar to previous attacks, allowing the ML system to identify them as patterns. This means that for ML to work, the data used for training and analyzing the network traffic must be good. Some problems still make ML seem too risky for NIDPS to use. Data volume is one issue, while irrelevant features are another [4, 5].

To solve these issues, choosing the right features is crucial for making ML-based NIDS work well. Choosing features not only reduces the amount of data we send to the model but also helps the model find the few but important attack patterns in network communications. In addition, selecting features aids in identifying crucial patterns and disregarding others, enabling efficient and effective data handling. In other words, feature selection improves accuracy and reduces training time by eliminating irrelevant or redundant features. By focusing on the most informative attributes, the model achieves better prediction performance while requiring fewer computational resources. We can use an optimization algorithm such as the Dragonfly Algorithm (DA) and the Bat Algorithm (BA) to determine the crucial set of features [6–9].

Combining ML and modern feature selection methods into NIDS is a big change in cybersecurity. By using effective algorithms to choose the best features and focusing on the most relevant data, cybersecurity experts

can build systems that are both flexible and efficient to secure communications over networks. They can also do it in a way that makes it difficult for attackers to find system weaknesses [9, 10]. Finally, the NIDS model we suggest uses several supervised ML classifiers, such as extreme gradient boosting (XGBoost), Random Forest (RF), Logistic Regression (LR), and Support Vector Machine (SVM) for intrusion detection. Each classifier has its strengths and weaknesses. The key contributions of this paper are; i) presenting a NIDPS framework that uses ML classifiers to efficiently handle large volumes of network data, supported by a new feature selection approach, ii) introducing an improved feature selection method that combines the DA and BA algorithms to eliminate irrelevant features and enhance detection accuracy, iii) evaluating the performance of the developed ML model using standard metrics such as accuracy, precision, recall, and F1-Score, iv) comparing the proposed model's effectiveness with existing attack detection models to highlight its strengths and improvements, and v) investigating different ML classifiers to identify the most suitable one for the proposed framework, with a focus on both detection performance and computational efficiency.

The remainder of this paper is organized as follows. Section II reviews related work. Section III presents the UNSW-NB15 dataset. Section IV describes the machine learning methods. Section V outlines the proposed approach. Section VI presents the implementation, results, and discussion. Finally, Section VII concludes the paper.

II. RELATED WORK

Ghasemi *et al.* [8] proposed an IDS system that is dedicated to handling Distributed Denial of Service (DDoS) attacks. The proposed IDS has used Harris Hawks Optimization (HHO) and DA algorithms for feature selection. The HHO and DA algorithms have independently selected candidate features. The selected feature by each algorithm is then merged using the Average Weighted Cutoff Method (AWCM) to retain the most relevant ones. The resulting subset of features is then refined using the SOPR local search technique to eliminate the redundant features. The Multilayer Perceptron (MLP) network is used to classify the attacks and benign data based on the final feature subset from HHO and DA algorithms. The proposed IDS system has achieved an accuracy of 96.9% and 84.5% with NSL-KDD and UNSW-NB15 datasets, respectively.

Farahani [9] implemented and compared the CFS and CFA feature selection algorithms to enhance the IDS performance. The CIC-IDS2017 has been used as a benchmark in the evaluation process. In addition, four of the well-known classifiers have been used in the training and testing of the selected subset of features—namely, SVM, NB, Decision Tree (DT), and K-Nearest Neighbor (KNN). With the CFA algorithm, the SVM, NB, DT, and KNN have achieved an accuracy of 96.02%, 94.59%, 96.87%, and 96.54%, respectively. With the CFS algorithm, the SVM, NB, DT, and KNN have achieved an accuracy of 97.12%, 96.32%, 97.91%, and 97.24%, respectively. Therefore, the DT has achieved the highest

performance with both CFA and CFS feature selection algorithms. However, the DT-CFS combination has outperformed the DT-CFA combination.

Safa *et al.* [10] proposed an IDS system that uses a hybrid method to improve the selection of the key attack features. The hybrid method combines CFS and Particle Swarm Optimization (PSO) algorithms. The CFS, as a feature evaluator based on PSO, is used to select the key features. After the NSL-KDD dataset is preprocessed, the proposed hybrid method has managed to reduce the number of features in the dataset from 41 to 21 feature. Then, the RF classifier is used to evaluate the performance of the proposed IDS with a hybrid feature selection method. The achieved accuracy with the new 21 subsets of features has reached 98.78%.

Kumar *et al.* [11] developed and validated an IDS system using the UNSW-NB15 dataset. The Information Gain (IG) algorithm forms the basis of this work's feature reduction strategy. The use of the IG algorithm caused the recognition of 22 features that are regarded as very important. Furthermore, the IDS that is presented in this study employed an integrated rule-based model that performed the classification process using multiple Tree-based classifiers. The performance of the proposed IDS system was assessed using the attack accuracy, the F-measure, and FAR. The proposed IDS achieved a high attack accuracy of 83.8%, an F-measure of 90%, and a low False Alarm Rate (FAR) of 2.01%.

Tama *et al.* [12] proposed a model of IDS that involves two phases of an ensemble method that consists of Rotation Forest and Bagging. A method of feature selection was used, which includes particle swarm optimization, ant colony optimization, and genetic algorithm. These algorithms have been applied to the UNSW-NB15 dataset, and the most beneficial 19 features were identified. To assess the efficiency of the proposed two-stage model, the researchers used the hold-out approach together with the 10-fold cross-validation. Some of the performance parameters that have been considered for this investigation are accuracy, False Positive Rate (FPR), sensitivity, and precision. The IDS that were established in this work achieved 91.27% accuracy, 91.3% sensitivity, and 91.6% precision.

Zong *et al.* [13] suggested an IDS that builds on the RF classifier using a two-phase method. The one-hot encoding method is applied to alter the nominal variables in the UNSW-NB15 dataset. The Information Gain (IG) method was employed to identify the most important features. The first phase of the method concentrates on determining classes that reflect a lack of representation concerning the entire population. Next, the second phase of the method continues to identify the classes that form the majority of the population. The method then incorporates the final state into the total prediction of the two phases to arrive at the ultimate decision. The performance measures the researchers used were accuracy and FAR. The proposed method underwent several experiments, achieving an accuracy level of 85.8% and a FAR of 15.8%.

The hybrid learning method underpins the selection of features constructed in Ref. [14]. The proposed IDS model

incorporates the selection and clustering of features. In the former case, the SVM is used, while in the latter case, a K-Medoids clustering algorithm is used. In addition, the method employs the NB classifier for performance evaluation using the KDD CUP99 dataset. In the assessment of the proposed model, three important parameters are considered: detection accuracy, detection rate, and alarm rate. The results of the experiment were then compared to the results of three other feature selection methodologies. The experiments reveal that the proposed hybrid normalization approach attains a higher accuracy of 91.5%, a detection rate of 90%, and a FAR of 6.4%.

Mohammadi *et al.* [15] developed a method for the selection of features for IDS systems. This method employs a blend of cluster techniques that combine the filter and wrapper algorithms. The filter algorithm applies a Cuttlefish Algorithm (CFA), whereas the wrapper algorithm applies a linear correlation coefficient algorithm (FGLCC). The performance of the proposed method is examined using the KDD CUP 99 dataset and the DT classifiers. The assessment results demonstrate that the proposed FGLCC-CFA method achieved a detection rate exceeding 95.23%, an accuracy of 95.03%, and a false acceptance rate of 1.65%.

Zhou *et al.* [16] able to successfully design an IDS system that selects the features using an ensemble approach. The framework uses both the BA algorithm and Correlation-Based Feature Selection (CFS) to select features. Furthermore, RF and forest-based penalizing attributes were applied when constructing the ensemble method. The present study used the CIC-IDS2017 dataset to analyze the performance of the CFS-BA method. The

efficiency of the CFS-BA method yields good performance with a detection rate of 94% and a t-false alarm rate of 2.4%.

Acharya and Singh [17] proposed a novel approach for selecting features for IDS systems based on the Intelligence Water Drops (IWD) mechanism. IWD is a bio-inspired algorithm that builds classifications using SVMs. This proposed novel approach was evaluated using the KDD CUP99 dataset. Experiments show that the IWD feature selection approach is highly efficient in detection rate (91.4%) and accuracy (93%) and has a minimal false alarm rate (3.3%).

III. UNSW-NB15 DATASET

The UNSW-NB 15 dataset was created at the University of New South Wales specifically to assess the NIDPSs. The dataset is relatively more recent than the other NIDPS datasets and contains modern benign and attack traffic. We chose the ISCX-URL2016 dataset for this study because it is well-known, widely used in attack detection research, and provides a good balance between attack and non-attack samples. Its richness and variety make it a reliable benchmark for evaluating the model's performance. The IXIA traffic generator has been used to capture 2,540,044 samples. Typically, a portion of 257,673 samples is used for the training and testing of the NIDPS. Nearly 61.6% of the samples are attacked, while the remaining 38.4% of samples are benign. The dataset contains 42 features divided into five types to represent the attack and benign samples. These features of the UNSW-NB 15 dataset are described in Table I [18–20].

TABLE I. FEATURES OF THE UNSW-NB 15 DATASET

#	Feature	Type	Description	Min. Value	Max. Value
1	dur	Numeric	Duration of the connection in seconds, representing how long the communication lasted.	0	59.996017
2	proto	nominal	Network protocol used in the connection, such as TCP, UDP, or ICMP.	N/A	N/A
3	service	nominal	Application-layer service associated with the connection, e.g., HTTP, FTP, DNS, SMTP.	N/A	N/A
4	state	nominal	Connection state at the time of capture, such as FIN (finished), CON (connected), or INT (interrupted).	N/A	N/A
5	spkts	Numeric	Total number of packets sent by the source host during the connection.	1	512
6	dpkts	Numeric	Total number of packets received by the destination host during the connection.	0	800
7	sbytes	Numeric	Total number of bytes sent from the source to the destination.	65	44196
8	dbytes	Numeric	Total number of bytes received by the destination from the source.	0	60800
9	rate	Numeric	Rate of transmitted packets per second, indicating the speed of the connection.	0	1000000
10	sttl	Numeric	Time-to-live (TTL) value of packets sent by the source, which helps track hops in the network.	31	255
11	dttl	Numeric	Time-to-live (TTL) value of packets received by the destination, indicating how far packets traveled.	0	254
12	load	Numeric	Average data transmission load of the connection, measured in bits per second.	0	2.224E+09
13	dload	Numeric	Average data load received by the destination, measured in bits per second.	0	818390.81
14	sloss	Numeric	Number of lost packets from the source, indicating possible congestion or packet drops.	0	2
15	dloss	Numeric	Number of lost packets at the destination, showing potential packet loss in transmission.	0	6
16	sinpkt	Numeric	Average time interval (in seconds) between consecutive packets sent by the source.	0	13992.212

#	Feature	Type	Description	Min. Value	Max. Value
17	dinpkt	Numeric	Average time interval (in seconds) between consecutive packets received by the destination.	0	13992.285
18	sjit	Numeric	Jitter (variation in packet arrival time) for source packets, which affects real-time communication.	0	19787.972
19	djit	Numeric	Jitter for packets received by the destination, showing delay variation in network traffic.	0	19788.074
20	swin	Numeric	TCP window size of the source, defining how much data can be sent before acknowledgment.	0	255
21	stcpb	Numeric	Sequence number of the first TCP packet sent by the source in the connection.	0	4277446941
22	dtcpb	Numeric	Sequence number of the first TCP packet received by the destination in the connection.	0	4088422545
23	dwin	Numeric	TCP window size of the destination, defining how much data can be received before acknowledgment.	0	255
24	tcprtt	Numeric	Estimated TCP round-trip time, measuring the time taken for a full handshake between source and destination.	0	0.265426
25	synack	Numeric	Time taken for the SYN-ACK phase of the TCP three-way handshake.	0	0.176175
26	ackdat	Numeric	Time taken for the ACK phase of the TCP handshake, completing the connection establishment.	0	0.136995
27	smean	Numeric	Mean (average) size of packets sent by the source, indicating data transmission characteristics.	50	834
28	dmean	Numeric	Mean (average) size of packets received by the destination.	0	865
29	trans_depth	Numeric	Number of pipelined requests in HTTP transactions, representing multiple requests within a connection.	0	131
30	response_body_len	Numeric	Length (in bytes) of the HTTP response body, excluding headers.	0	5242880
31	ct_srv_src	Numeric	Count of connections from the same source IP to a particular service, showing repeated access.	1	59
32	ct_state_ttl	Numeric	Number of connections with the same state and TTL, used for traffic pattern analysis.	0	6
33	ct_dst_ltm	Numeric	Number of connections targeting the same destination IP over time.	1	59
34	ct_src_dport_ltm	Numeric	Number of times a source IP connected to a particular destination port.	1	59
35	ct_dst_sport_ltm	Numeric	Number of times a destination IP received traffic from a particular source port.	1	38
36	ct_dst_src_ltm	Numeric	Number of times the same source and destination IPs communicated.	1	59
37	is_ftp_login	Numeric	Boolean (1 = Yes, 0 = No), indicating whether an FTP session included a login attempt.	0	1
38	ct_ftp_cmd	Numeric	Number of FTP commands issued during the session, useful for detecting brute force attacks.	0	2
39	ct_flw_http_mthd	Numeric	Count of HTTP request methods (e.g., GET, POST) observed in the connection.	0	16
40	ct_src_ltm	Numeric	Number of connections from the same source IP over time.	1	60
41	ct_srv_dst	Numeric	Number of times a particular service was accessed on the destination IP.	1	59
42	is_sm_ips_ports	Numeric	Boolean (1 = Yes, 0 = No), indicating whether the source and destination have identical IPs and ports, useful for detecting certain attacks.	0	1

IV. OVERVIEW OF ML METHODS

This section presents a review of the supervised ML classifiers employed in this work, namely XGBoost, SVM, RF, and LR. The study focuses on traditional ML classifiers due to their efficiency, interpretability, and suitability for real-time or resource-limited intrusion detection environments. While deep learning models (e.g., Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks) are powerful, they often require more data and computational resources. Traditional classifiers, when combined with effective feature selection like the DAuBA approach, can still deliver highly accurate and practical results.

A. XGBoost

XGBoost is a powerful ensemble learning algorithm that builds decision trees sequentially while correcting previous errors using gradient boosting. It excels in attack detection due to its exceptional accuracy in handling high-dimensional data, imbalanced datasets, and complex attack patterns. Its regularization techniques prevent overfitting, ensuring robust generalization in cybersecurity applications. Additionally, XGBoost's parallel processing and scalability make it highly efficient for large-scale NIDSs [21, 22].

B. SVM

SVM is an ML classifier that is capable of performing both classification and regression operations, and it is very

well-known in ML research. The goal of the SVM classifier is to find the ideal hyperplanes by first dividing the dataset that is being used into a number of different groups. One of the many benefits of utilizing SVM is the fact that it tends to perform well even when presented with high-dimensional input spaces. In addition to this, the decision-making process can use any number of Kernel functions thanks to the flexibility offered by the SVM classifier. One of its drawbacks is that it needs precise tuning in most situations, but this is especially true when the input dimension is higher than the number of Ref. [23, 24].

C. RF

RF is an ensemble learning algorithm that constructs multiple decision trees and aggregates their outputs for improved accuracy and robustness. In attack detection, RF is highly effective due to its ability to handle high-dimensional data, resistance to overfitting, and strong

generalization capabilities. It performs well in identifying complex attack patterns, even in noisy or imbalanced datasets. Additionally, its parallel processing capability makes it scalable for large-scale NIDSs [21, 25].

D. LR

Despite the fact that it is referred to as “regression,” LR is an ML classifier that is most frequently used for the problem of binary classification. The LR can also be utilized for multiclass classification tasks, in which case the learning algorithm will use the one-vs-rest method to complete the task. When applied to a linear ML method, the sigmoid function or one of its many versions is used in the LR model. This operation will provide output that is compressed between [0 and 1]. The likelihood of a particular class can be determined by looking at an output that is closer to 1. Table II shows the main aspects of the XGBoost, RF, SVM, and LR classifiers [21, 24].

TABLE II. MAIN ASPECTS OF THE XGBOOST, RF, SVM, AND LR CLASSIFIER

Aspect	XGBoost	RF	SVM	LR
Learning Approach	Ensemble, Boosting	Ensemble, Bagging	Supervised, Maximum Margin	Supervised, Statistical
Working Mechanism	Uses gradient boosting to optimize weak learners sequentially	Builds multiple decision trees and aggregates their outputs	Finds an optimal hyperplane that separates attack and normal instances	Uses a logistic function to model the probability of an instance belonging to an attack class
Strengths	High accuracy, handles large datasets, robust against overfitting	Handles high-dimensional data, reduces variance, robust against noise	Effective for high-dimensional data, works well with small datasets	Simple, interpretable, performs well on linearly separable data
Weaknesses	Computationally expensive, requires hyperparameter tuning	Slower for large datasets, prone to overfitting with too many trees	Computationally expensive for large datasets, sensitive to noise	Assumes linearity, struggles with complex non-linear relationships
Performance on Attack Detection	High accuracy and generalization, effectively detects sophisticated attacks	Strong generalization, reduces false positives	Effective at detecting well-separated attacks, but may struggle with overlapping classes	Works well for simple attack patterns but may struggle with complex non-linear threats
Scalability	Highly scalable but computationally intensive	Scalable but requires more memory with large datasets	Less scalable for large datasets due to quadratic complexity	Highly scalable for large datasets with linear decision boundaries
Suitability for NIDS with UNSW-NB15	Very effective due to boosting capabilities and ability to handle imbalanced data	Performs well on high-dimensional network traffic data	Useful for detecting attack patterns with clear separation	Suitable for basic attack detection but may require transformation techniques for complex data

V. THE PROPOSED APPROACH

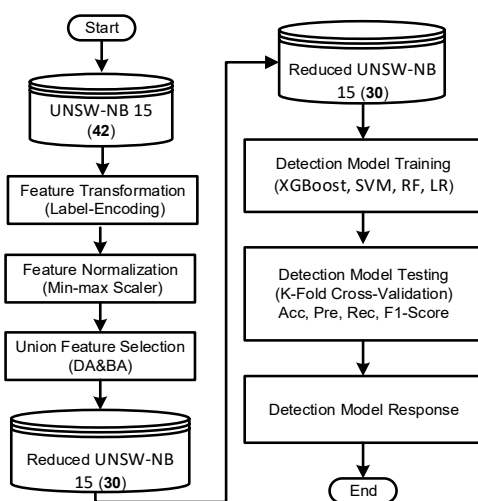


Fig. 1. Architectural of the proposed NIDPS system.

Fig. 1 details the architectural design that was proposed for the system that was investigated in this study. The first block is dedicated entirely to the processing of the data. This stage is essential to the completion of a fruitful learning process. The processing of data consists of three steps: the transformation step, the normalization step, and the feature selection step. The selection of features is carried out with the aid of the DAuBA algorithm. Following the selection of the necessary feature vector, the model is trained with the training set. After that, the trained model is put to the test by employing the test dataset. The technique that was just described is carried out iteratively until a tuned and suitable model is located.

A. Feature Transformation

Generally, ML classifiers are using mathematical computations to classify the data. Therefore, the input to the ML classifiers must be numbers. Feature transformation techniques, such as Label-Encoding, are

used to make sure that the values of the features are numbers. The Label-Encoding technique assigns a unique integer to each value in the feature. For instance, assuming the number of values in the feature is n , then the Label-Encoding technique assigns an integer between 0 and n to each value [26, 27].

B. Feature Normalization

The large number of a variety of features affects the learning process for ML classifiers such as XGBoost, SVM, RF, and LR. This is because these methods all take these values into account. Additionally, the training of high-dimensional datasets needed a significant amount of computer resources. Data is frequently scaled by employing techniques such as max normalization, Z-decimal scaling, and min-max scaling. This is done in an effort to reduce the impact of the problems mentioned above. In this step of our data processing, we make use of the min-max scaling (Eq. (1)) [26, 27].

$$\text{Min - Max scaling of feature } F: F_{norm} = \frac{F - F_{min}}{F_{max} - F_{min}} \quad (1)$$

C. Feature Selection

The performance of ML algorithms, particularly for classification tasks, relies heavily on the provision of a low-dimensional representation of raw data. A discriminative feature set must be used to represent the input raw data in order to streamline ML models, make interpretations easier, save training time, lessen the bad effects of dimensionality, and stop overfitting. Frequently, this process is known as feature selection. Feature selection is an essential ML step for data pre-processing because of the continuously growing amounts of data. Experts have proposed numerous strategies for feature selection. In the last decade, metaheuristic strategy algorithms have shown dominance among feature selection strategies. Metaheuristic algorithms are highly effective for addressing feature selection [28, 29]. Metaheuristic algorithms perform several tasks to achieve effective feature selection. Fig. 2 presents the schematic representation of the primary tasks carried out by metaheuristic algorithms in a general flowchart design. The tasks illustrated in Fig. 2 are explained below:

Step 1. Generate Initial Population

Randomly generate a set of candidate solutions (called the population).

Each solution represents a possible answer to the optimization problem.

Step 2. Calculate Fitness Value (Initial Evaluation)

Evaluate how good each solution is using a fitness function.

The fitness function measures how well the solution solves the problem.

Step 3. Metaheuristic Search (Exploration & Exploitation)

Apply the metaheuristic algorithm (e.g., PSO, GA, DA, BA, etc.) to guide the search.

Exploration: Search new areas of the solution space.

Exploitation: Refine known good solutions.

Step 4. Calculate Fitness Value (After Search)

Recalculate fitness for the newly generated or modified solutions.

Step 5. Update Population

Replace old solutions with better ones based on fitness.

Step 6. Repeat steps 3–5 until a maximum number of iterations is reached

Step 7. Return the best-performing solution found during the process.

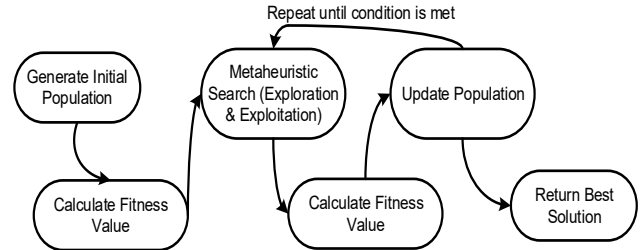


Fig. 2. Flowchart of metaheuristic algorithms.

1) The BA and DA algorithm

This study presents a novel approach for selecting features by utilizing the widely recognized BA and DA metaheuristic optimization algorithms. The BA and DA algorithms enhance NIDPS by optimizing feature selection. Thus improving the detection of attacks in inbound and outbound network traffic. The BA efficiently identifies the most relevant network features and removes redundant data by simulating bat echolocation. Its strong global search and adaptive tuning help detect evolving cyber threats with high precision. The DA algorithm also enhances NIDPS by optimizing feature selection. The DA effectively balances exploration and exploitation and identifies the most critical features while eliminating irrelevant data by simulating dragonfly swarming behavior. The adaptive search mechanism in both algorithms enables real-time identification of evolving threats. Therefore, they are highly effective for securing dynamic network environments. Table III compares and contrasts the BA and DA algorithms [6, 7, 30, 31].

2) Union of BA and DA algorithms

Union feature selection combines multiple feature selection methods to create a more robust and comprehensive set of features. The union feature selection approach brings together multiple techniques — filter, wrapper, and embedded methods — to identify a more comprehensive set of relevant features. This combination helps uncover important patterns that individual methods might miss. By drawing on the strengths of each technique, the model becomes more robust, less prone to overfitting, and achieves higher accuracy on unseen data. It mitigates the individual weaknesses and biases of each method, ensuring a more balanced and reliable feature set. Furthermore, combining methods can lead to more efficient training and prediction times, as well as enhanced model interpretability by highlighting the most significant features from multiple perspectives. This comprehensive approach ultimately results in a more effective and trustworthy model.

TABLE III. COMPARES AND CONTRAST THE BA AND DA ALGORITHMS

Criteria	BA algorithm	DA algorithm
Inspired By	Echolocation behavior of bats	Swarming behavior of dragonflies
Optimization Strategy	Frequency tuning, velocity updates, and pulse emission	Attraction, repulsion, and alignment phases
Exploration vs. Exploitation Balance	Strong exploitation; limited exploration	Balanced exploration and exploitation
Feature Reduction	Moderate reduction, retains more features	Higher reduction, removes more redundant features
Convergence Speed	Slower due to frequent local optima trapping	Faster with a more efficient search mechanism
Classification Accuracy	High accuracy but slightly lower than DA	Higher accuracy due to better feature selection
Computational Cost	Higher due to increased function evaluations	Lower due to effective exploration-exploitation balance
Robustness	Sensitive to parameter tuning	More stable across different feature subsets

The proposed union feature selection method balances exploration and exploitation by combining BA and DA. The BA algorithm contributes strong exploitation capabilities through its local search behavior, guided by echolocation-inspired adjustments in frequency and pulse rate. This allows it to fine-tune feature subsets effectively. In contrast, the DA algorithm enhances exploration by simulating social interaction dynamics such as alignment, separation, and cohesion, helping the search avoid premature convergence. By uniting these two algorithms, the method benefits from DA’s wide search coverage and BA’s precision, improving the overall quality of the selected features. This balance is essential to avoid local optima and achieve a robust feature subset.

The DA and BA algorithms were chosen over the PSO-Genetic Algorithm (PSO-GA) hybrid because they offer a better balance between performance and efficiency in feature selection. PSO-GA methods often struggle with issues like early convergence in PSO and the high processing cost of GA due to its use of crossover and mutation. In contrast, DA is good at exploring the search space widely, while BA focuses on fine-tuning solutions through its echolocation behavior. By combining these two, the proposed approach can select important features more effectively and with less computational effort. This makes the DAuBA hybrid more practical and suitable for real-time intrusion detection compared to PSO-GA. The proposed DAuBA union feature selection method scales reasonably well to large datasets due to its adaptive search mechanisms and parallel feature subset generation. However, as with most population-based metaheuristics, computational complexity may increase with higher dimensionality.

The union set theory function in mathematics forms the foundation of the proposed feature selection approach. The union of selected features from DA and BA was adopted to capture a wider range of relevant attributes. Since DA and BA follow different search strategies, each may select unique but complementary features. Using the union helps retain this diversity. In contrast, relying on the intersection could exclude important features and reduce detection performance. In addition, the intersection achieved accuracies of 98.1%, 98.4%, 97%, and 96.5% with the XGBoost, RF, SVM, and LR classifiers, respectively. However, the union of the algorithms outperforms their intersection, as demonstrated in Section VI. The union function combines the elements of two or more sets without duplication. First, using the essential features from

the UNSW-NB15 dataset, the DA algorithm generates the first subset. The BA method then generates a second subset from the UNSW-NB15 dataset based on important feature selection. At last, depending on the Union function, the elements of the DA and BA subsets are merged into one set. Fig. 3 illustrates the proposed DA \cup BA features selection method.

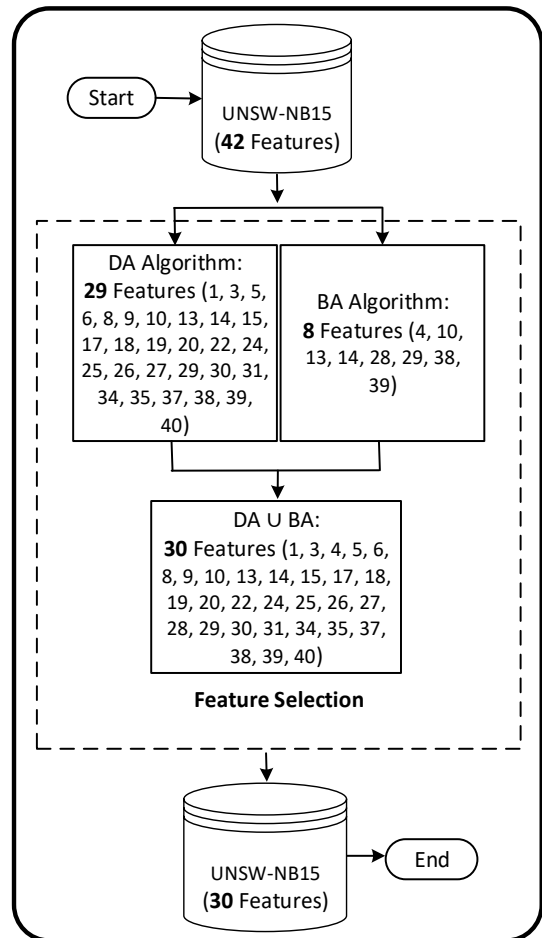


Fig. 3. The suggested DA \cup BA features selection technique.

VI. IMPLEMENTATION, RESULT, AND DISCUSSION

The suggested NIDPS model tests have been conducted on a VivoBook S13 S333JQ laptop with the following specifications: The operating system is Windows 10 Pro for business, the CPU is Intel Core i7-1065 G7 (1.3 GHz quad-core with Turbo Boost (up to 3.9 GHz) and 8MB

cache), the graphic is NVIDIA GeForce GPU (2 GB GDDR5 VRAM), the memory is 6 GB 2666 MHz DDR4, and the storage 512 GB PCIe SSD. Python has also been used for implementation purposes. Python contains several libraries to implement the required ML algorithms. Some used libraries are MinMaxScaler, LabelEncoder, RandomForestClassifier, XGBClassifier, OneClassSVM, mealpy.swarm_based.BA, mealpy.swarm_based.DA, train_test_split, and accuracy_score.

The values of the BA algorithm's hyperparameters are population size (pop_size) = 50, minimum frequency (f_min) = 0.0, maximum frequency (f_max) = 2.0, loudness ($loudness$) = 0.5, pulse emission rate ($pulse_rate$) = 0.5, loudness update factor (α) = 0.9, and pulse rate update factor (γ) = 0.9. On the other hand, the values of the DA algorithm's hyperparameters are population size (pop_size) = 50, separation weight (c_s) = 0.1, alignment weight (c_a) = 0.1, cohesion weight (c_c) = 0.1, attraction to food (c_f) = 2.0, distraction from enemy (c_e) = 1.0, inertia weight (w) = 0.9, inertia damping factor (w_damp) = 0.99, maximum step size (δ_max) = 0.1, and minimum step size (δ_min) = 0.0.

TABLE IV. EVALUATION METRICS

Metric	Purpose	Strengths	Weaknesses	Equation
Accuracy	Measures the proportion of correctly classified instances (both attack and normal) out of the total instances.	Simple to compute and interpret; useful when class distribution is balanced.	Can be misleading in imbalanced datasets where one class dominates.	$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$
Precision	Measures how many of the instances predicted as attacks are actually attacks.	Useful when false positives need to be minimized (e.g., reducing false alarms in NIDS).	Can be misleading if false negatives are high, as it does not consider missed attacks.	$Precision = \frac{TP}{(TP + FP)}$
Recall	Measures how many actual attacks are correctly detected.	Essential in security applications where missing an attack is costly.	High recall may come at the expense of increased false positives.	$Recall = \frac{TP}{(TP + FN)}$
F1-Score	Harmonic mean of Precision and Recall, balancing both metrics.	Provides a balanced evaluation when both false positives and false negatives matter.	Less interpretable than individual metrics; does not consider true negatives.	$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$

B. Result and Discussion

XGBoost, SVM, LR, and RF were the ML algorithms that were taken into consideration as part of our experimental design. In the experiments, we utilized the reduced feature space (30 attributes) of the UNSW-NB15 dataset for the binary configuration. The outcomes of our experimental procedures are presented in the Figs. 4–7, where the figures are presenting the results achieved by the ML classifiers for the binary classification. To assess the generalization performance of the proposed model, 5-fold cross-validation was employed. The dataset was randomly partitioned into five equal subsets, where each subset was used once as the test set while the remaining four served as the training set. Figs. 4–7 show the accuracy, recall, precision, and the F1-Score, respectively.

A. Performance Criteria

There are a number of criteria that can be used to assess ML-based NIDS systems; however, the objective of this research is to achieve the highest possible percentage of accurate predictions for the instances contained in the test dataset. The Accuracy, as shown in Table IV, should be the primary criterion that you look at. Where “TP” stands for “True Positive” and refers to the percentage of instances that are accurately identified as “attacks.” TN, which stands for True Negative, refers to the percentage of instances that are accurately identified as “un-attacks.” FP, which stands for false positive and can also be referred to as Type I error, refers to the percentage of instances that are inaccurately identified as “attacks.” FN, which stands for false negative and can also be referred to as Type II error, refers to the percentage of instances that are inaccurately identified as “un-attacks.” In this paper, we also take into consideration the recall metric, the precision metric, and the F1-Score, which is defined in Table IV [32–35].

In the XGBoost algorithm, the $n_estimators$, max_depth , $learning_rate$, min_child_weight , γ , $subsample$, $colsample_bytree$, reg_alpha , and reg_lambda were set to 100, 6, 0.3, 1, 0, 1.0, 1.0, 0, and 1, respectively. In the RF algorithm, the $n_estimators$, max_depth , $max_features$, $min_samples_split$, $min_samples_leaf$, and n_jobs were set to 100, None, sqrt, 2, 1, and None, respectively. Using these settings, the XGBoost and RF algorithms achieved accuracy, recall, precision, and an F1-Score of 100%. The results indicate that the 30 features selected by the proposed union feature selection methods are perfect for the XGBoost and RF algorithms. While the XGBoost and RF classifiers achieved 100% accuracy on the test dataset, this result should be interpreted with caution. Such performance, although encouraging, may indicate potential overfitting or dataset-specific bias. The UNSW-

NB15 dataset used in this study is well-structured and relatively balanced, which may contribute to higher accuracy. Moreover, the proposed DA-BA feature selection method reduces dimensionality and enhances class separability, further boosting classification performance. However, no model can be expected to generalize perfectly in all real-world scenarios.

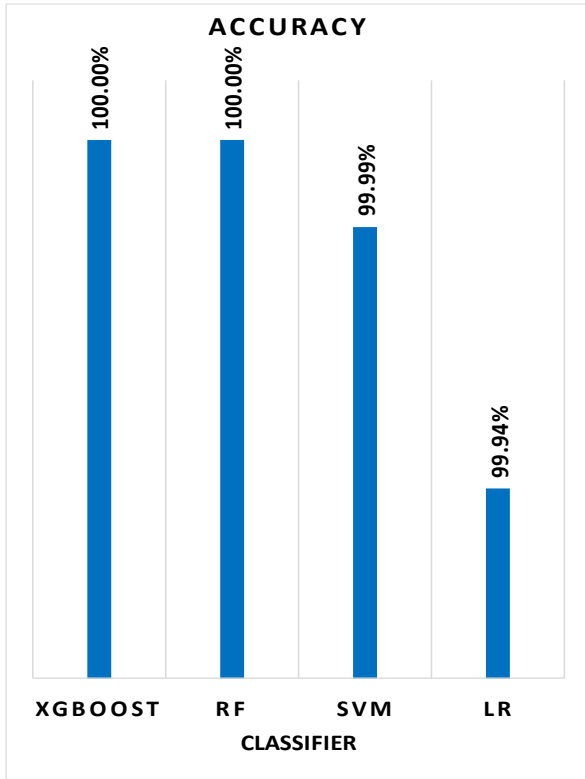


Fig. 4. Accuracy of the proposed NIDPS system.

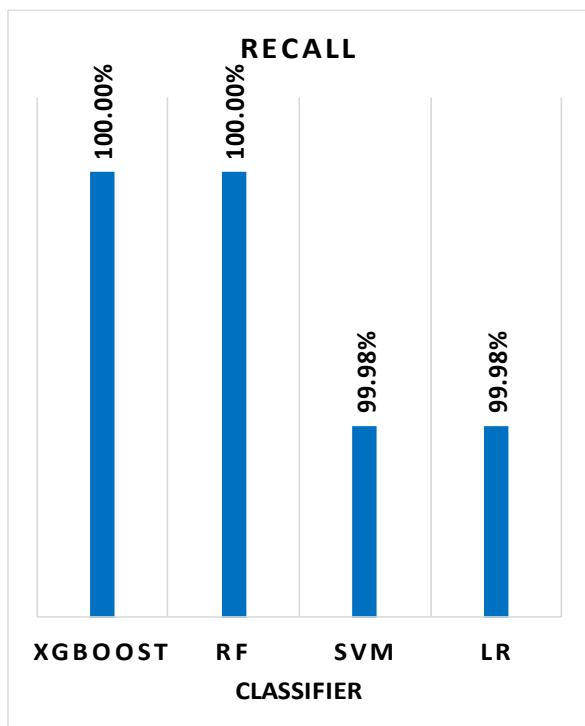


Fig. 5. Recall of the proposed NIDPS system.

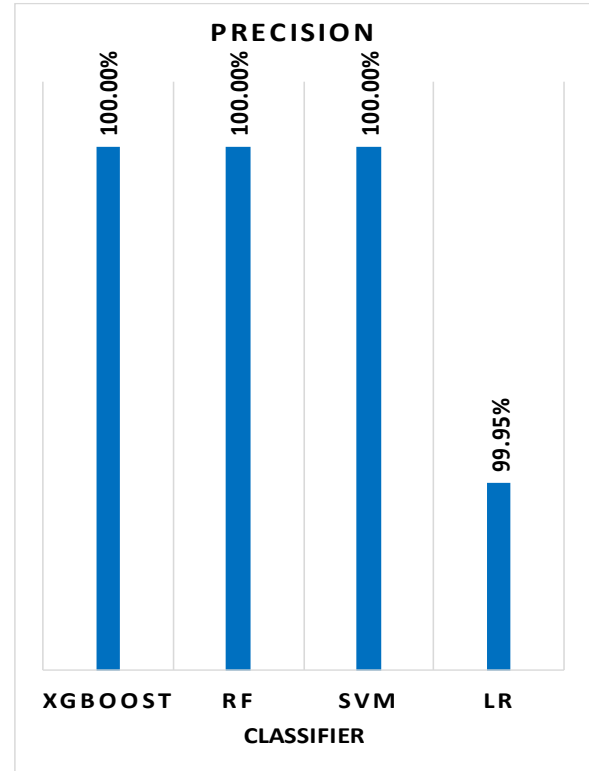


Fig. 6. Precision of the proposed NIDPS system.

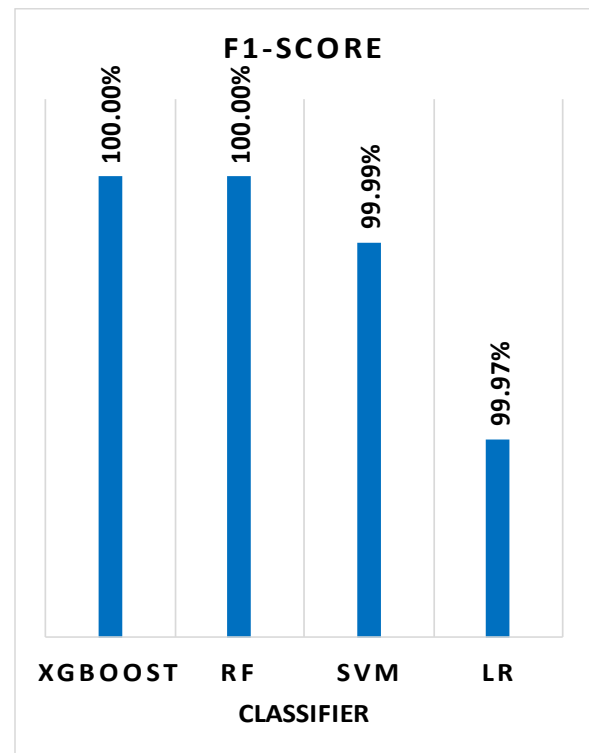


Fig. 7. F1-Score of the proposed NIDPS system.

In the SVM algorithm, the C, kernel, gamma, and class_weight were set to 1.0, rbf, scale, and None, respectively. In the LR algorithm, the penalty, C, solver, max_iter, and class_weight were set to l2, 1.0, l bfgs, 100, and None, respectively. Using these settings, the SVM algorithm achieved accuracy, recall, precision, and an F1-Score of 99.99%, 99.98%, 100.00%, and 99.99%,

respectively. In addition, the LR algorithm achieved accuracy, recall, precision, and an F1-Score of 99.94%, 99.98%, 99.95%, and 99.97%, respectively. The results indicate that the 30 features selected by the proposed union feature selection methods are nearly perfect for the SVM and LR algorithms.

Fig. 8 shows the accuracy of the proposed NIDPS system compared to the previous works. The proposed work in Ref8, Ref9, Ref10, Ref12, Ref14, Ref15, and Ref17 has achieved an accuracy of 96.9%, 97.91%, 98.78%, 91.3%, 91.5%, 95.03%, and 93%, respectively. On the other hand, the suggested NIDPS system has attained an accuracy of 100%, 100%, 99.99%, and 99.94% with XGBoost, RF, SVM, and LR classifiers, respectively. Therefore, the proposed NIDPS system has substantially outperformed the previous works with all classifiers. Accordingly, the settings of the proposed NIDPS system, including the proposed union features election method and the tuned classifiers, are promising to secure communications over the network.

Figs. 9–12 present the confusion matrices for XGB, SVM, LR, and RF, respectively. These matrices offer a more detailed and objective assessment of classifier performance under the proposed DAuBA feature selection framework. Both XGB and RF achieved ideal results, with zero FP and FN, demonstrating excellent precision and recall—crucial metrics in intrusion detection tasks. SVM recorded 2 FN and no FP, indicating high precision but a slight drop in sensitivity due to a few undetected attack instances. LR, while still effective, produced 6 FP and 2 FN, reflecting reduced precision and recall in comparison. These results suggest that the DAuBA-selected features are highly discriminative, especially when used with ensemble-based classifiers like XGB and RF.

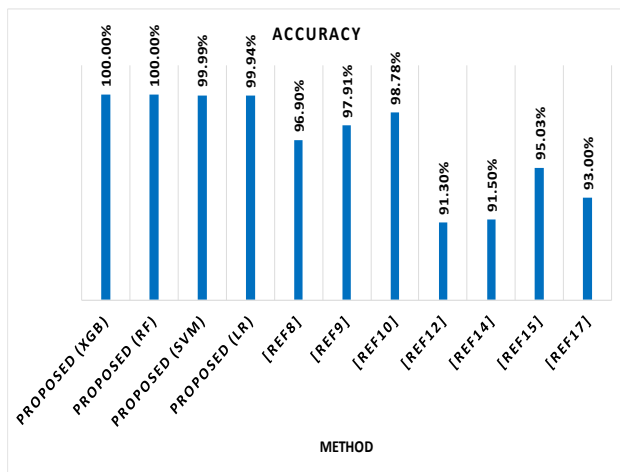


Fig. 8. Accuracy of the proposed NIDPS system compared to the previous works.

TP=11714	FN=0
FP=0	TN=2019

Fig. 9. XGBoost Confusion matrix.

TP=11712	FN=2
FP=0	TN=2019

Fig. 10. SVM Confusion matrix.

TP=11714	FN=0
FP=0	TN=2019

Fig. 11. RF Confusion matrix.

TP=11712	FN=2
FP=6	TN=2013

Fig. 12. LR Confusion matrix.

C. Computational Efficiency and Runtime Analysis

The proposed DAuBA-based feature selection framework introduces an additional offline optimization phase. This step, however, does not impact real-time detection, as it is performed prior to model training. By reducing the original feature space from 42 to 30 features (a 28.57% reduction), the framework lowers computational complexity and improves training and inference efficiency across classifiers. The time complexity of the feature selection phase can be approximated as $O(P \cdot D \cdot T)$, where P is the population size, D is the number of features, and T is the number of iterations. Although this process adds to the preprocessing time, the reduced feature set significantly enhances model performance and scalability.

TABLE V. TRAINING TIME

Classifier	Training Time (42 Features)	Training Time (30 Features)	Time Reduction (%)
XGB	1.80 sec	1.30 sec	27.78%
RF	2.10 sec	1.52 sec	27.62%
SVM	3.60 sec	2.55 sec	29.17%
LR	0.85 sec	0.61 sec	28.24%

To evaluate the impact on runtime, the average training times of four classifiers — XGB, RF, SVM, and LR — were measured using both the original and reduced feature sets. As shown in Table V, all classifiers demonstrated a notable decrease in training time, ranging from approximately 27% to 29%. The reduction was especially significant for SVM and XGB, which are more sensitive to input dimensionality. Furthermore, confusion matrix results indicate that this computational gain does not come at the cost of predictive performance; false alarm rates remained extremely low (e.g., 0% for XGB and RF and 0.3%

for LR), demonstrating the effectiveness of the proposed feature selection strategy in supporting real-time and resource-constrained applications.

VII. CONCLUSION

In order to develop precise NIDSs, this research investigated the use of a union algorithm for feature selection in conjunction with various ML techniques, including XGBoost, SVM, RF, and LR. The effectiveness of these techniques was evaluated using the UNSW_NB15 dataset. The binary classification setting was taken into consideration in this work. Additionally, the UNSW_NB15 dataset was subjected to the union of DA and BA feature selection approach, and as a result, 30 optimal features have been chosen. To put our findings into context, we compared the performance results acquired using our suggested methodology to those produced using a summary of the performance outcomes achieved by the numerous algorithms in the literature. We performed the experiments utilizing the decreased feature vector generated by the union of DA and BA feature selection approaches. The outcomes of the experiment showed that utilizing a reduced feature vector provides advantages in terms of improving the detection of attacks. The XGBoost and RF classifiers achieved an accuracy of 100% in identifying the network intrusion, which proves the efficiency of the proposed union feature selection method with these classifiers.

The proposed DA-BA feature selection method has shown promising results; however, several limitations remain. The method's performance is sensitive to hyperparameter settings, which were manually selected and may not generalize well across different datasets. Additionally, the evaluation was conducted on a single benchmark dataset, limiting the assessment of its broader applicability. The current study also does not include empirical comparisons with other hybrid metaheuristic approaches such as PSO-GA. To address these limitations, future work will focus on analyzing hyperparameter sensitivity, exploring adaptive tuning techniques, applying the method to additional datasets with cross-validation, and benchmarking it against other state-of-the-art hybrid optimization algorithms.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

M.M.A. and S.N.A., conceptualized the study and designed the methodology; M.Z. and I.Q., conducted the experiments and performed data analysis; M.O.H., contributed to results interpretation and validation; S.M.A., reviewed the manuscript and provided critical revisions; all authors had approved the final version.

REFERENCES

- [1] T. Harada, T. Shimada, and T. Yoshida, "Dynamic mobile X-haul reconfiguration by fast network switching for low-latency MEC service," *IEICE Commun. Express*, vol. 13, no. 12, pp. 500–503, Dec. 2024. doi: 10.23919/comex.2024XBL0135

- [2] M. M. Abualhaj, S. A. Khatib, M. O. Hiari, and Q. Y. Shambour, "Enhancing spam detection using hybrid of Harris Hawks and Firefly optimization algorithms," *J. Soft Comput. Data Min.*, vol. 35, no. 2, pp. 161–174, 2024. doi: 10.30880/jscdm.2024.05.02.012
- [3] H. Benaddi, K. Ibrahim, A. Benslimane, M. Jouhari, and J. Qadir, "Robust enhancement of intrusion detection systems using deep reinforcement learning and stochastic game," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 11089–11102, Oct. 2022. doi: 10.1109/TVT.2022.3186834
- [4] H. A. Mimi, N. A. Hamad, M. M. Abualhaj, S. N. A. Khatib, and M. O. Hiari, "Improved intrusion detection system to alleviate attacks on DNS service," *J. Comput. Sci.*, vol. 19, no. 12, pp. 1549–1560, Dec. 2023. doi: 10.3844/jcssp.2023.1549.1560
- [5] B. Gao, B. Bu, W. Zhang, and X. Li, "An intrusion detection method based on machine learning and state observer for train-ground communication systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6608–6620, Jul. 2022. doi: 10.1109/TITS.2021.3058553
- [6] S. Mirjalili, "Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Comput. Appl.*, vol. 27, no. 4, pp. 1053–1073, May 2015. doi: 10.1007/s00521-015-1920-1
- [7] P. W. Tsai, J. S. Pan, B. Y. Liao, M. J. Tsai, and V. Istanda, "Bat algorithm inspired algorithm for solving numerical optimization problems," *Appl. Mech. Mater.*, vol. 148–149, pp. 134–137, Dec. 2011. doi: 10.4028/www.scientific.net/AMM.148-149.134
- [8] J. Ghasemi, R. Salah-hassan, and K. G. Firouzjah, "A combined harris hawks and dragonfly optimization approach for feature selection in MLP-based DDoS attack detection," *International Journal of Engineering*, vol. 38, no. 8, pp. 1898–1908, 2025. doi: https://doi.org/10.5829/ije.2025.38.08b.14
- [9] G. Farahani, "Feature selection based on cross – Correlation for the intrusion detection system," *Secur. Commun. Netw.*, vol. 2020, no. 1, 8875404, 2020.
- [10] B. Safa, R. Mohamed-Hamou, and A. Toumouh, "Optimizing the performance of the IDS through feature-relevant selection using PSO and random forest techniques," *Computation Sistemas*, vol. 28, no. 2, pp. 473–488, 2024.
- [11] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, "An integrated rule-based intrusion detection system: Analysis on UNSW-NB15 data set and the real time online dataset," *Chust. Comput.*, Oct. 2019. doi: 10.1007/s10586-019-03008-x
- [12] B. A. Tama, M. Comuzzi, and K. H. Rhee, "TSE-IDS: a two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," *IEEE Access*, vol. 7, pp. 94497–94507, 2019. doi: 10.1109/ACCESS.2019.2928048
- [13] W. Zong, Y.-W. Chow, and W. Susilo, "A two-stage classifier approach for network intrusion detection," *Lect. Notes Comput. Sci.*, pp. 329–340, Jan. 2018. doi: 10.1007/978-3-319-99807-7_20
- [14] L. Khalvati, M. Keshtgary, and N. Rikhtegar, "Intrusion detection based on a novel hybrid learning approach," *J. Artif. Intell. Data Min.*, vol. 6, no. 1, pp. 157–162, Mar. 2018. doi: 10.22044/jadm.2017.979
- [15] S. Mohammadi, H. Mirvaziri, M. G. Ahsae, and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," *J. Inf. Secur. Appl.*, vol. 44, pp. 80–88, Feb. 2019. doi: 10.1016/j.jisa.2018.11.007
- [16] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Comput. Netw.*, vol. 174, 107247, Jun. 2020. doi: 10.1016/j.comnet.2020.107247
- [17] N. Acharya and S. Singh, "An IWD-based feature selection method for intrusion detection system," *Soft Comput.*, vol. 22, no. 13, pp. 4407–4416, May 2017. doi: 10.1007/s00500-017-2635-2
- [18] O. Almomani, A. Alsaaidah, A. A. Abu-Shareha, A. Alzaqebah, M. A. Almaiah, and Q. Shambour, "Enhance URL defacement attack detection using particle swarm optimization and machine learning," *J. Comput. Cogn. Eng.*, Feb. 2025. doi: 10.47852/bonviewjce52024668
- [19] J. Xie, H. Wang, J. M. Garibaldi, and D. Wu, "Network intrusion detection based on dynamic intuitionistic fuzzy sets," *IEEE Trans. Fuzzy Syst.*, vol. 30, no. 9, pp. 3460–3472, Sept. 2022. doi: 10.1109/TFUZZ.2021.3117441

- [20] P. Verma, J. G. Breslin, D. O'Shea, N. Mehta, N. Bharot, and A. Vidyarthi, "Leveraging gametic heredity in oversampling techniques to handle class imbalance for efficient cyberthreat detection in IIoT," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 1940–1951, Feb. 2024. doi: 10.1109/TCE.2023.3319439
- [21] L. A. Dabbas *et al.*, "Early detection of female type-2 diabetes using machine learning and oversampling techniques," *Journal of Applied Data Sciences*, vol. 5, no. 3, pp. 1237–1245, 2024.
- [22] R. Elnaggar, L. Servadei, S. Mathur, R. Wille, W. Ecker, and K. Chakrabarty, "Accurate and robust malware detection: Running XGBoost on runtime data from performance counters," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 41, no. 7, pp. 2066–2079, Jul. 2022. doi: 10.1109/TCAD.2021.3102007
- [23] A. A. A. Shareha, "A framework for diabetes detection using machine learning and data preprocessing," *J. Appl. Data Sci.*, vol. 5, no. 4, pp. 1654–1667, Dec. 2024. doi: 10.47738/jads.v5i4.363
- [24] Q. Shambour, "Artificial intelligence techniques for early autism detection in toddlers: A comparative analysis," *J. Appl. Data Sci.*, vol. 5, no. 4, pp. 1754–1764, Dec. 2024. doi: 10.47738/jads.v5i4.353
- [25] M. M. Abualhaj, Q. Y. Shambour, A. A. A. Shareha, S. N. A. Khatib, and A. Amer, "Enhancing malware detection through self-union feature selection using gray wolf optimizer," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 37, no. 1, p. 197, Jan. 2025. doi: 10.11591/ijeecs.v37.i1.pp197-205
- [26] M. Madi, F. Jarghon, Y. Fazea, O. Almomani, and A. Alsaaidah, "Comparative analysis of classification techniques for network fault management," *Turk. J. Electr. Eng. Comput. Sci.*, vol. 28, no. 3, pp. 1442–1457, May 2020. doi: 10.3906/elk-1907-84
- [27] M. A. Siddiqi and W. Pak, "An agile approach to identify single and hybrid normalization for enhancing machine learning-based network intrusion detection," *IEEE Access*, vol. 9, pp. 137494–137513, 2021. doi: 10.1109/ACCESS.2021.3118361
- [28] M. M. Abualhaj, M. O. Hiari, A. Alsaaidah, and M. M. A. Zyoud, "Comparative analysis of whale and harris hawks optimization for feature selection in intrusion detection," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 37, no. 1, pp. 179–185, Jan. 2025. doi: <https://doi.org/10.11591/ijeecs.v37.i1.pp179-185>
- [29] M. M. Abualhaj, S. A. Khatib, N. A. Shafi, I. Qaddara, and AHyassat, "Utilizing gray wolf optimization algorithm in malware forensic investigation," *Journal of Computational and Cognitive Engineering*, May 2025. doi: <https://doi.org/10.47852/bonviewjcece52025053>
- [30] K. Taji, A. Sohail, T. Shahzad, B. S. Khan, M. A. Khan, and K. Ouahada, "An ensemble hybrid framework: A comparative analysis of metaheuristic algorithms for ensemble hybrid CNN features for plants disease classification," *IEEE Access*, vol. 12, pp. 61886–61906, 2024. doi: 10.1109/ACCESS.2024.3389648
- [31] W. A. H. M. Ghanem *et al.*, "Cyber intrusion detection system based on a multiobjective binary bat algorithm for feature selection and enhanced bat algorithm for parameter optimization in neural networks," *IEEE Access*, vol. 10, pp. 76318–76339, 2022. doi: 10.1109/ACCESS.2022.3192472
- [32] M. A. Almaiah, L. M. Saqr, L. A. A. Rawwash, L. A. Altellawi, R. Al-Ali, and O. Almomani, "Classification of cybersecurity threats, vulnerabilities and countermeasures in database systems," *Comput. Mater. Continua*, vol. 81, no. 2, pp. 3189–3220, Nov. 2024. doi: 10.32604/cmc.2024.057673
- [33] Y. Sanjalawe, S. Fraihat, S. Al-E'Mari, S. Makhadmeh, and E. Alzubi, "A Review of 6G and AI convergence: enhancing communication networks with artificial intelligence," *IEEE Open Journal of the Communications Society*, vol. 6, pp. 2308–2355, 2025. doi: 10.1109/OJCOMS.2025.3553302
- [34] Y. Sanjalawe, S. Fraihat, M. Abualhaj, S. R. A. E. Mari, and E. Alzubi, "Hybrid deep learning for human fall detection: A synergistic approach using YOLOv8 and time-space transformers," *IEEE Access*, vol. 13, pp. 41336–41366, 2025. doi: 10.1109/ACCESS.2025.3547914
- [35] M. M. Abualhaj, A. A. A. Shareha, M. O. Hiari, Y. Alrabanah, M. A. Zyoud, and M. A. Alsharaiah, "A paradigm for DoS attack disclosure using machine learning techniques," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 3, Jan. 2022. doi: <https://doi.org/10.14569/ijacsa.2022.0130325>

Copyright © 2025 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).