Future Technology
Open Access Journal
ISSN 2832-0379

May 2025| Volume 04 | Issue 02 | Pages 41- 50

Journal homepage: https://fupubco.com/futech

https://doi.org/10.55670/fpll.futech.4.2.5

Future Publishing LLC

Article

An innovative ZFFinch-MLPNet architecture for improving cyber intrusion prediction efficiency and accuracy

Dinesh Kumar Budagam*

Sr cybersecurity engineer at VISA, Inc, Foster City, CA, United States

ARTICLE INFO

Article history:
Received 28 February 2025
Received in revised form
04 April 2025
Accepted 16 April 2025

Keywords:
Intrusion detection system, Cyber security, Exploratory data analysis, Recursive feature elimination, MLPRNN classification, Zebra-Falcon finch algorithm

*Corresponding author
Email address:
dbudagam@gmail.com

DOI: 10.55670/fpll.futech.4.2.5

ABSTRACT

Intrusion Detection System (IDS) is one of the most significant security elements in today's information technology-related organizations. For overcoming intrusion detection difficulties, Deep Learning (DL) has shown a significant contribution in recent times. An innovative IDS that merges the Zebra- Falcon Finch algorithm with a Multi-Layer Perceptron Recurrent Neural Network (ZFFinch-MLPNet) classifier is developed in this research. To assure data compatibility and integrity, the developed work initiates with data preprocessing comprising data inspection, handling missing values, and label encoding. Then, to recognize the structure of data, the Exploratory Data Analysis (EDA) combines correlation and visualization analysis. To enhance the intrusion detection efficacy, a Recursive Feature Elimination (RFE) is utilized to select the appropriate features. Finally, the MLPRNN classification approach with the Zebra-Falcon Finch algorithm offers flexibility, opposing overfitting, and improved accuracy. Also, for detecting network anomalies, this work addresses the developed approach's outcomes in Python software and compares it with modern approaches. It is confirmed that the developed approach detects distinct types of network intrusions and attains better performance in identification with an accuracy of 96.20%, MCC of 92.33%, and ROC of 0.99.

1. Introduction

Recently, the requirement for defenses and cybersecurity against cyberattacks has risen. Robbing online bank accounts, divulging sensitive data through public channels, such as intruding on the crucial infrastructure of the nation, and infecting systems with malware are some of the cyberattacks. Loss of information or data is a significant effect of cyberattacks worldwide [1, 2]. In the digital world, it is important to protect critical infrastructure and sensitive data opposing cyberattacks. Due to enhancements in cyberattacks and refined attack vectors, conventional security approaches are incapable of keeping up with recent threats [3]. IDSs are most effective and vital in defense against hostile attacks on infrastructures and network resources by analyzing user behavior and network traffic [4]. Conversely, numerous attacks that emerge in tandem with the rapid expansion of the amount of data are hacked and are accessible online. Despite their effectiveness, the conventional NIDS combat to conserve with the evolving approaches of cyberattacks [5]. Due to the serious difficulty of modern cyberattacks, more novelty is required while implementing strong IDS to protect crucial assets and network data by classifying and detecting network traffic and finding rare activity [6].

Thus, integrating Artificial Intelligence (AI) into IDS enhances network security. For improving the modern network devices, a DL based IDS is exploited. In particular, DL applied to network logs in a business network discovers patterns in the data and uses patterns to detect possible attacks as irregularities in network traffic. Combining the knowledge from many administrative areas greatly improves the performance of such models [7, 8]. In Valente and Maldonado's [9] research, a forward feature selection approach is developed, which is more interpretable by selecting a subset of features, which is more easily analyzed and understood. However, the misinteractions between features occur as it evaluates features individually rather than in combination. The Backward elimination for feature selection aids in detecting the most significant features, leading to a more interpretable model. Nevertheless, this method evaluates each feature individually, potentially overlooking interactions between features [10]. Therefore, the Recursive Feature Elimination-based feature selection is implemented to focus on the most important features, leading to better predictions. The classification algorithms are exploited to detect cyber-attacks, including Support Vector Machine (SVM) [11], which is exploited to find malicious

Abbreviations

ABC	Artificial Bee Colony
AI	Artificial Intelligence
ARP	Address Resolution Protocol
AUC	Area Under Curve
DL	Deep Learning
DoS	Denial of Service
DT	Decision Tree
EDA	Exploratory Data Analysis
FA	Firefly Algorithm
IDS	Intrusion detection system
KNN	K Nearest Neighbor
MCC	Matthews Correlation Coefficient
MLP	Multi-Layer Perceptron
NIDS	Network Intrusion Detection system
OSPF	Open Shortest Path First
PSO	Particle Swarm Optimization
RFE	Recursive Feature Elimination
RF	Random Forest
RNN	Recurrent Neural Networks
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
SVM-GMM	Support Vector Machine with Gaussian Mixture Model
UDP	User Datagram Protocol
UNAS	Universal Network Access System
ZFFinch-MLPNet	Zebra- Falcon Finch algorithm with a Multi-Layer Perceptron Recurrent Neural Network

behavior within low-rate, power, and small-range networks. The difficulty of the SVM is to detect the support vectors when classifying unidentified traffic as either benign or malicious. The Random Forest (RF) algorithm was developed in [12] to address the issue of inadequate capabilities of IDS. It generates a different number of decision trees from diverse samples and takes their majority vote for the classification decision. The advantage of RF is that improved precision is achieved without overfitting, but depending on the specific data and parameters used, there is still a risk of balancing bias and variance.

A neural network with a straightforward structure is the Multi-Layer Perceptron (MLP), introduced by M. Alazab et al. [13]. This indicates that, in comparison to other sophisticated DL techniques, it generates precise detection results in a promising amount of running time. The model becomes more complex as the number of neurons and layers rises, making training and fine-tuning more challenging. The internet's security is strengthened through the use of convolutional neural networks [14]. By classifying all network packet traffic into malicious or benign classes, this IDS model seeks to identify network intrusions. However, it requires large amounts of labeled training data, high computational costs, and the potential for overfitting. Recurrent Neural Networks (RNNs) are capable of briefly remembering previous states and applying them to the current computation [15, 16]. While RNNs are good at a variety of prediction tasks, they have a vanishing gradient issue. These issues are overcome by utilizing the Multi-Layer Perceptron Recurrent Neural Network classifier.

The performance of the developed classifier is enhanced by optimization algorithms like Particle Swarm Optimization (PSO) [17], Genetic Algorithm [18], Firefly Algorithm (FA) [19], and Artificial Bee Colony (ABC) algorithm [20]. However, these approaches have a slow convergence rate, are challenging to encode complex problems, are less effective, and require careful parameter tuning. To overcome such issues, this research develops a Zebra-Falcon finch-optimized Multi-Layer Perceptron Recurrent Neural Network classifier. The main objectives are:

- Integrating the preprocessing approach for enhancing data integrity and compatibility.
- Utilizing Exploratory Data Analysis to enhance its detection accuracy by revealing vital information about the data.
- Incorporating the Recursive Feature Elimination-based feature selection aiding in detecting the most vital features.
- Exploiting MLPRNN classifier to accurately identify the cyber-attacks, which is optimized via a Zebra-Falcon finch algorithm, provides enhanced accuracy and resilience against overfitting.

2. Methodology

The developed intrusion detection system's block diagram is depicted in Figure 1. Input data, which is collected from a UNSW_ NB15 dataset, contains inconsistencies, noise, and missing values. Thus, the preprocessing contains three stages: Data inspection, which analyzes the quality and structure of data; handling missing values, which ensures completeness by utilizing an imputation or removal technique; and Label encoding, which transforms categorical data into numerical form for better processing of the classifier. Then, the EDA aids in understanding data distribution and detecting anomalies from the pre-processed data.

To enhance the performance, the RFE approach is exploited, where less important features are removed to retain the relevant features that reduce the dimensionality and enhance the model efficacy. Then, the data undergoes scaling and normalization, where data scaling standardizes feature values to the same range, and data normalization assures consistent distribution and averts biased weights among features. This prepares the dataset for model training and testing. To enhance the performance, the Zebra-Falcon Finch optimization algorithm is utilized, which fine-tunes the hyperparameters. Finally, the MLPRNN classifier is utilized where the MLP is appropriate for structured classification tasks, and RNN manages the sequential data effectively, thereby enhancing security and protecting sensitive information.

2.1 Preprocessing

To effectively represent the quality of data, the UNSW_ NB15 dataset needs to be pre-processed. The dataset has been pre-processed, undergoing stages like data inspection, handling missing values, and label encoding.

2.1.1 Data inspection

In preprocessing, data inspection is the process of analyzing data before it is subjected to additional analysis or transformation. This stage is crucial to guarantee data quality, detect problems, and comprehend the dataset's properties.

2.1.2 Handling missing values

One data preprocessing method for producing a smooth dataset is missing value management. Determining whether the dataset has any missing values is the first step in the process. There are several ways to deal with missing values: ignore those values completely, replace them with any number, replace them with the attribute's mean value, or replace them with the value that appears the most frequently for that feature. The features' mean or mode values are used to fill in the missing data.

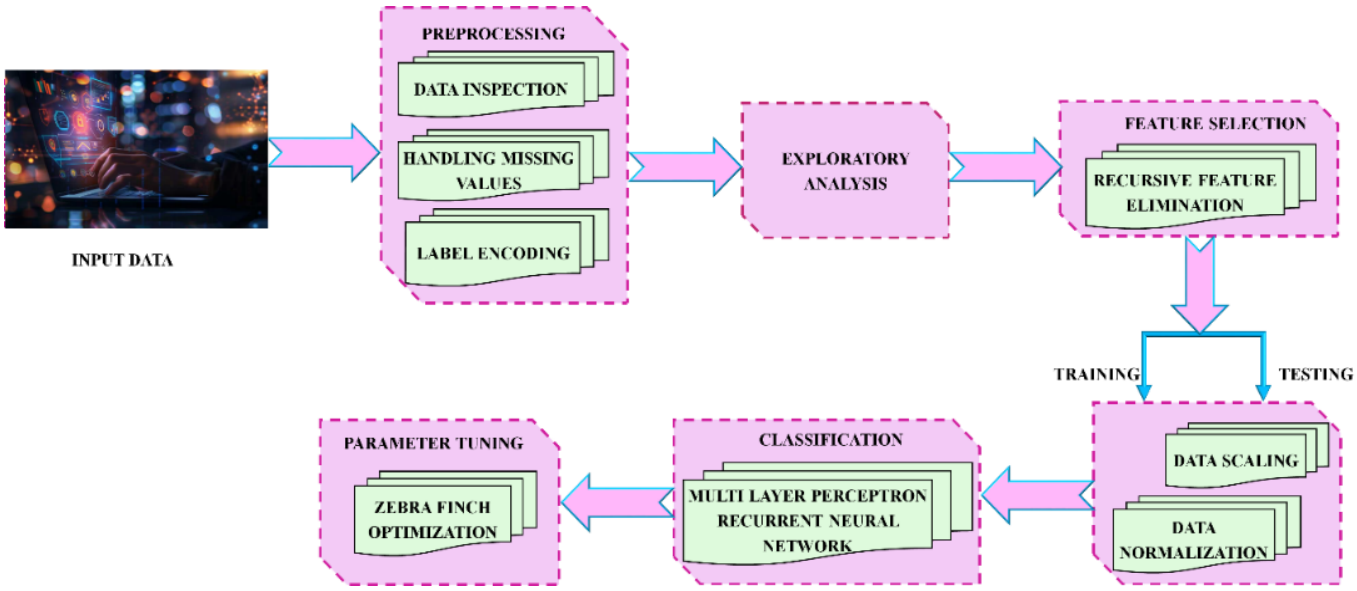

Figure 1. The proposed block diagram

2.1.3 Label encoding

The process of transforming category data into numerical values so that DL algorithms can use them is known as label encoding. To facilitate the process in the training phase, categorical values are converted into numerical demonstrations to train a DL model. This is achieved by substituting integers between 0 and (n-1), where n' is the total number of distinct classes for categorical values. Then, the EDA is exploited to detect patterns and anomalies discussed below.

2.2 Exploratory data analysis

The vital process of performing preliminary data exploration is detecting patterns and anomalies, testing hypotheses, and validating assumptions with the assistance of summary statistics and graphical representations EDA. It analyzes the types of recorded assaults and allows for a general understanding of cybersecurity attacks, as seen in Figure 2.

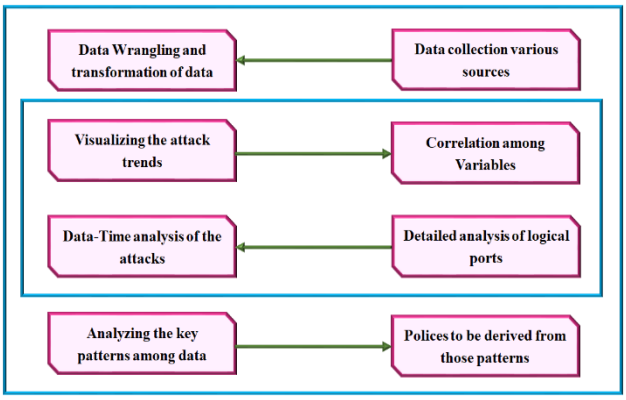

Figure 2. Structure of EDA

2.2.1 Correlation among the variables

Relationships between the variables are discovered during these phases. Pearson's correlation is utilized to find a linear relationship among the variables, while Spearman correlation is exploited to detect a monotonic relationship. Heat maps are used to illustrate the data distribution, and variable pairs are used to identify associations.

2.2.2 Date time analysis of cybersecurity attacks and detailed analysis of logical ports

A thorough examination of the attack's date and time is conducted in order to identify trends and comprehend the timing of the attacker's tactics. A scatterplot is created for analysis, with each attack representing a point associated with the destination port. The analysis is done on how the source and destination logical ports behaved during the cyberattacks.

2.2.3 Summarizing statistics by hypothesis testing

The results are tested using hypothesis testing to determine whether significant findings emerge. A statistical test to determine whether the means of the two groups differ from one another (whether the mean of the cybersecurity assaults' source ports differs from the mean of their destination ports) is reasonable. The ability to find whether something occurred, whether particular treatments have beneficial effects, whether groups vary from one another, or whether one variable forecasts another makes hypothesis testing one of the most vital ideas. Subsequently, the RFE-based feature selection is utilized to select the relevant features.

2.3 Recursive feature elimination-based feature selection

By removing the less important elements, feature selection reduces training time and improves learning performance, all of which contribute to the creation of a more accurate model. Following preprocessing, the correlation between features and the percentage of each feature that is positively and negatively correlated are determined. Because of its simplicity of usage and design, as well as its ability to effectively pick features in training datasets that are pertinent to predicting target variables and eliminating weak features, the Recursive Feature Elimination (RFE) algorithm is highly popular. By identifying a strong link between particular

features and targets (labels), the RFE approach is used to choose the most important features.

A base estimator is utilized to detect the impact of each feature in the current training set. The low-priority features are then eliminated to develop a new subset of features. RFE's fundamental principle is to repeat this recursive procedure on a new subset of features until the desired number of features has been chosen. It is necessary to ascertain the base estimator, the number of features chosen (n features to choose), and the feature removal step when using RFE for feature selection. The relevance of each feature (feature parameters) is determined using an optimal base estimate based on the training set. Until a feature set is obtained that matches the number of features elected, the number of features deleted during each recursion is managed in accordance with the feature removal step. The final feature selection differs based on the feature removal phase, the base estimator, or the base estimator's parameters, even if the number of features chosen remains constant. Finally, the Zebra-Falcon finch-optimized MLPRNN classifier is used for accurately detecting cyber-attacks.

2.4 Zebra- Falcon Finch optimized MLPRNN classifier

In order to process sequential data and capture temporal dependencies, an MLPRNN is a neural network architecture that combines the standard structure of MLP with recurrent connections. This gives the network a memory of previous inputs within a sequence. An input layer, one or more hidden layers, and an output layer are the feed-forward components of a standard MLPRNN network. On the other hand, trainable feedback connections with a unit time delay are used to link the nodes of a particular layer of an MLPRNN. A generic MLPRNN design with feedback connections in all hidden layers and the output layer is illustrated in Figure 3. In order to give the network the proper context, the inputs contain the target output's previous value. The following difference equations are used to define the equations describing the j^{th}node at the k^{th}layer of an RMLP network at the time n:

$$v_{kj}(n) = \sum_{i=0}^{P_{k-1}} w_{kj,i} \cdot y_{k-1,i}(n) + \sum_{i=1}^{Pf_k} wf_{kj,i} \cdot y_{kj}(n-1) \quad (1)$$

$$y_{kj}(n) = \varphi_k\left(v_{kj}(n)\right) \quad (2)$$

Where $y_{kj}(n)$ is the output signal from the node, and $v_{kj}(n)$ is the internal state variable of the j^{th} node at the k^{th}layer. $w_{kj,i}$stands for the network forward connection weight that connects the node $i's$ output at the layer $k-l$ to node $j's$ input at layer $k, wf_{kj,i}$stands for the network feedback weight that connects node i's output at layer k to node j's input at the same layer, the total number of layer inputs applied to node j is denoted byP_{k-1}, while the total number of layer k feedback inputs applied to node j is denoted by Pf_k. The nonlinear sigmoidal activation function at layer k is denoted by $\varphi_k(\cdot)$. Figure 3 represents the flowchart of the Zebra-Falcon finch-optimized MLPRNN classifier.

The threshold applied to the node j at layer k is equal to the weight $w_{kj}(n)$, which corresponds to the fixed bias input of -1. The weighted total of feedback from the layer k at time $n-1$ and the weighted sum from the preceding layer $k-l$ at time n make up the contributions of the internal state variable $v_{kj}(n)$ of the j^{th} node at the k^{th} layer. The MLPRNN develops a completely recurrent connection at the layer k when the number of feedback, $Pf_k$is identical to the total number of outputs at the layer k is P_k. There are no feedback connections for the MLPRNN at that layer if $Pf_k$is equal to zero. The Zebra-Falcon Finch optimization algorithm is a bio-inspired optimization approach utilized to enhance parameter tuning in the MLPRNN classifier.

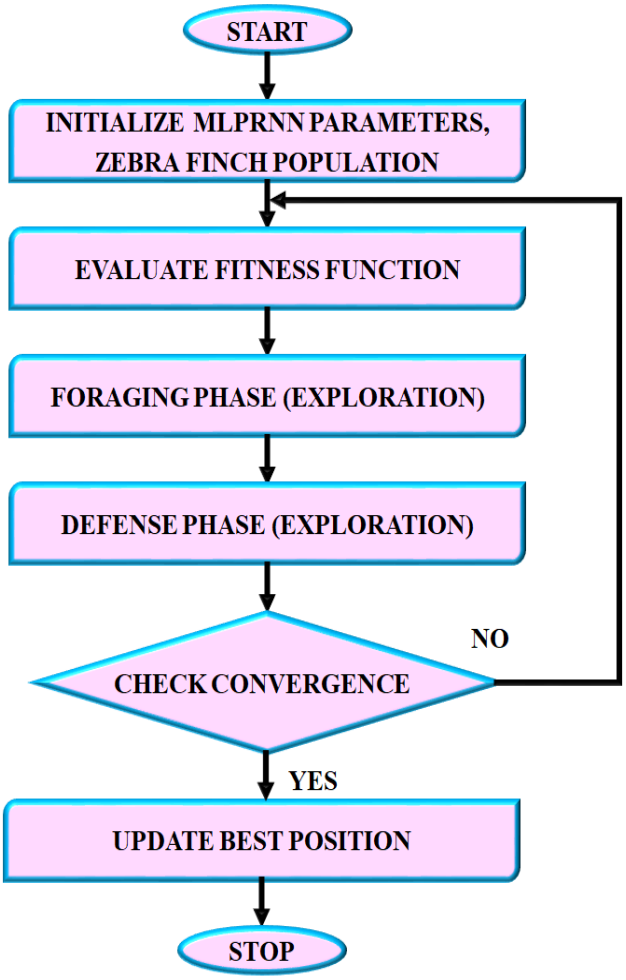

Figure 3. Flowchart of Zebra-Falcon finch optimized MLPRNN classifier

It is inspired by the foraging and defensive strategies of zebra Falcon finches, and it incorporates Levy flight for exploration and self-adaptive population adjustment for effective convergence. The behavior of zebra finches, especially for foraging and defensive strategies against predators, forms the basis for the search mechanism of zebra Falcon finches. The optimization process is initiated by initializing a population of fitches that indicate the distinct set of parameters (biases and weights of MLPRNN). Each Zebra-Falcon finch is denoted by

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,m} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,m} \end{bmatrix} \quad (3)$$

Where the number of parameters being optimized is m and the number of Zebra-Falcon fitches in the population is N. Each candidate solution is estimated based on its fitness function, which is defined by

$$F_i = f(X_i) \quad (4)$$

Where the fitness function applied to each Zebra-Falcon finch is cap F sub i., and the present position of the Zebra-Falcon finch is subscript base, Debra-Falcon finch is subscript base,

ebra-Falcon finch is X_i. The optimization process follows a foraging behavior, where Zebra-Falcon finches explore the search space. The best individuals (producers) lead the exploration, while others (scroungers) follow. The position of the Zebra-Falcon finches is updated using,

$$x_{i,j}^{new,P1} = x_{i,j} + r.\left(PZ_j - I.x_{i,j}\right) \quad (5)$$

Where the position of the best Zebra-Falcon finch is PZ_j, the integer value is I and the random number is r. The new position is accepted only if it enhances the objective function is:

$$X_i = \begin{cases} X_i^{new,P1} F_i^{new,P1} < F_i \\ X_i \quad , \quad Otherwise \end{cases} \quad (6)$$

This phase encourages the algorithm to explore the search space more widely and avoid getting stuck in the local optima. Then, the defense phase simulates the reaction of Zebra-Falcon finches to predator attacks. It incorporates escape and gathering strategies. The updated rule for the defense phase is:

$$x_{i,j}^{new,P2} = \begin{cases} x_{i,j} + R.(2r-1).\left(1-\frac{t}{T}\right)x_{i,j}, & P_s \le 0.5 \\ x_{i,j} + r.\left(AZ_j - I.x_{i,j}\right), & Otherwise \end{cases} \quad (7)$$

Where the maximum number of iterations is T, attacked Zebra-Falcon finch's position isAZ_j, the current iteration is t and the probability of selecting among escape or gathering is P_s. If it enhances the objective function, the new position is updated by:

$$X_i = \begin{cases} X_i^{new,P2} F_i^{new,P2} < F_i \\ X_i \quad , \quad Otherwise \end{cases} \quad (8)$$

This phase enhances the exploration capability and helps to refine the solution after exploration. The Zebra-Falcon Finch optimization algorithm is a powerful optimization technique that enhances MLPRNN training by balancing exploration and exploitation**,** making it highly effective in neural network parameter tuning**.**

3. Results and Discussions

The developed cyber intrusion detection system is implemented in Python software, and its results are included in this section. Also, the comparison with conventional approaches is incorporated to reveal the proficiency of the developed research.

Figure 4 demonstrates the distribution of data for attack and normal categories. The attack class constitutes 55.06%, whereas the normal class constitutes 44.94%. The attack type indicates detected malicious activities like illegal access and malware attacks. Also, the normal type represents legitimate network traffic that enables expected behavior patterns. The imbalance denoted in the chart highlights the significance of improving detection accuracy and reducing false positives. If the attack rate is consistently high, it shows vulnerabilities in the system's security architecture, and better encryption and anomaly-based detection methods. This chart is used by security analysts to monitor attack patterns over time, adjust response strategies, and strengthen overall system resilience.

The attack category distribution in the network traffic is represented in Figure 5. It denotes the frequency of different types of network intrusions alongside normal network activity. The chart highlights that Normal network activity is the most frequent category, with a value of 37,000 instances. The most common attack category is Generic with 18,871 instances, indicating that general, non-specific attacks make up a significant portion of malicious activity. Exploits with 11,132 instances, representing attempts to take advantage of system vulnerabilities.

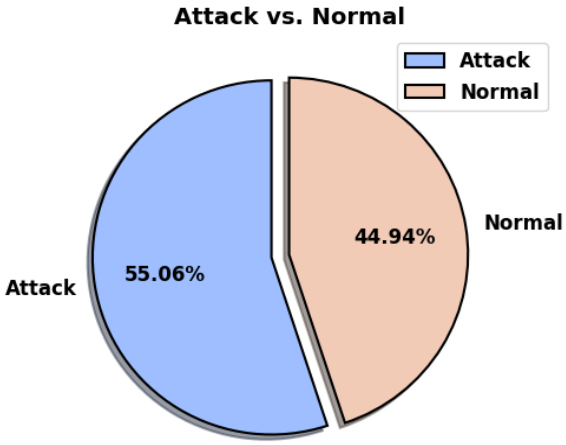

Figure 4. Distribution of data

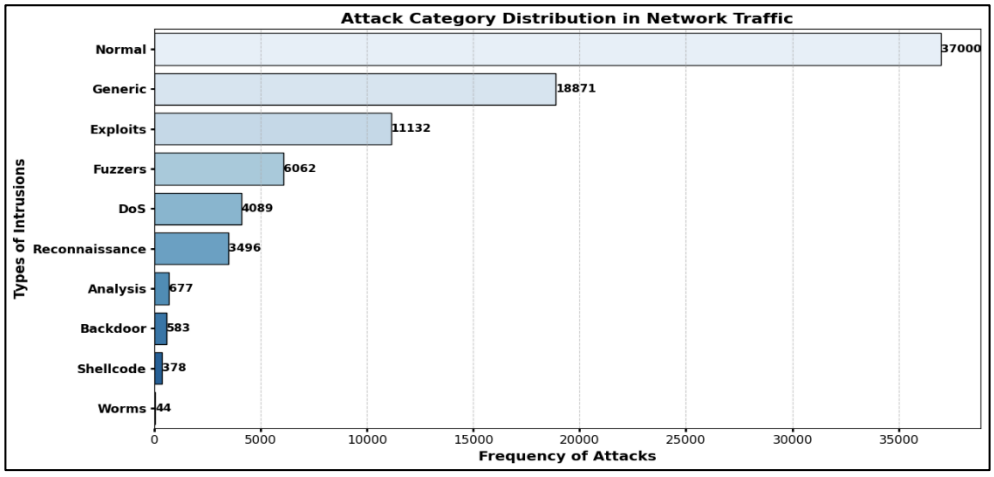

Figure 5. Attack category distribution in the network traffic

Fuzzers have 6,062 instances and are automated tools designed to test system faults by injecting malformed data. Denial of Service (DoS) attacks account for 4,089 instances, reflecting attempts to disrupt service by overwhelming the network with traffic. The chart also shows Reconnaissance attacks with 3,496 instances, which involve gathering information about a system to identify vulnerabilities. Analysis attacks, which involve a detailed examination of network traffic or system behavior, appear less frequently with 677 instances. More targeted attacks, such as Backdoor, which has 583 instances, and Shellcode, which has 378 instances, are relatively rare but potentially very dangerous since they allow attackers to bypass security measures and execute malicious code. Worms, which are self-replicating programs that spread across systems, are the least frequent, with only 144 instances. The high frequency of Generic and Exploit attacks denotes that many attackers rely on automated tools and known vulnerabilities to compromise systems. Security teams use this data to prioritize defense strategies, focusing on the most frequent and dangerous attack types while improving system resilience.

The top 5 protocols in network traffic are indicated in Figure 6. It illustrates the frequency of different network protocols observed in network traffic, which is crucial information for intrusion detection. The most frequently used protocol is Transmission Control Protocol (TCP), with a significant value of 43,095 instances, indicating that most of the network traffic relies on a connection-oriented protocol that ensures reliable data delivery. User Datagram Protocol (UDP) follows with 29,418 instances, reflecting the high volume of connectionless traffic, which is often used for real-time applications like streaming. The third most common protocol is the Universal Network Access System (UNAS), with 3,515 instances representing an uncommon or application-specific protocol. Address Resolution Protocol (ARP) appears with 987 instances, indicating its role in mapping IP addresses to MAC addresses within a local network. Open Shortest Path First (OSPF) is the least frequent among the top five, with 676 instances. High UNAS traffic denotes non-standard or proprietary protocols being exploited for data exfiltration or covert communication. The data in the chart highlights the significance of tracking both common and uncommon protocols to enhance network security and quickly respond to threats.

Figure 7 signifies the training and validation results for IDS. These graphs are typically exploited to calculate the performance of a DL model, particularly in intrusion detection. In the accuracy graph, both the training and validation accuracy start at around 0.6 and rapidly improve during the first 25 epochs. After that, the validation accuracy continued to increase gradually and stabilized at 0.9620 by the 100th epoch, indicating that the model is learning effectively and generalizing well to the validation data. After that, the validation loss gradually declined and stabilized at 0.0976. In intrusion detection, these outcomes denote that the model has been well-trained and is capable of accurately distinguishing between normal and malicious network traffic.

The low loss value proves that the detection model is reliable and does not exhibit overfitting, whereas the great accuracy denotes that the developed approach detects patterns in network traffic linked with attacks. These outcomes are better for IDS, denoting that the approach is accurate while exploited in real-time scenarios.

Figure 8 reveals the confusion matrix that estimates the performance of a DL model used for intrusion detection. From an intrusion detection standpoint, the high number of True Positives and True Negatives indicates that the model is highly accurate in detecting both attacks and normal traffic. The small number of False Positives (356) means that the model has a low false alarm rate, which is crucial for reducing preventable alerts. However, the presence of 546 False Negatives indicates that some attacks still bypass detection, which is a vulnerability in the model. This confusion matrix confirms that the model is highly effective for intrusion detection and shows promise for real-world deployment in network security monitoring.

The Receiver Operating Characteristic (ROC) curve is displayed in Figure 9. The curve rises sharply near the origin and quickly levels off near the top, representing that the model attains high recall with a low false positive rate. The Area Under Curve (AUC) value is 0.99, which reflects near-perfect classification performance. An AUC value close to 1.0 specifies that the model is extremely proficient in differentiating between normal and attack traffic with very few misclassifications. Such a high AUC value also indicates that the model is not overfitting or underfitting, as it performs consistently well across different thresholds.

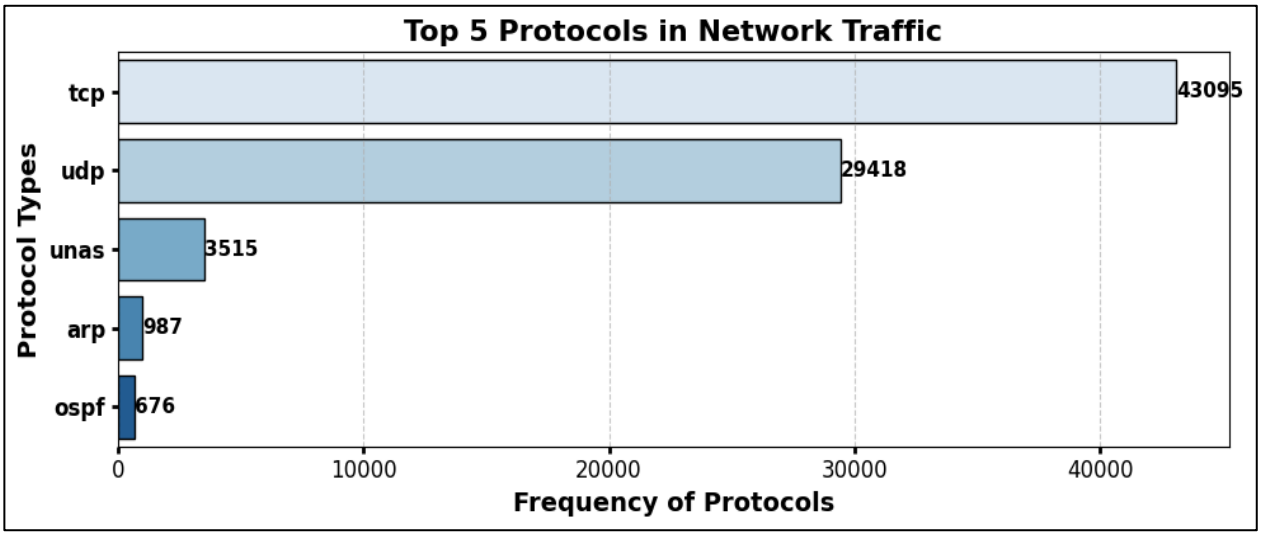

Figure 6. Top 5 protocols in network traffic

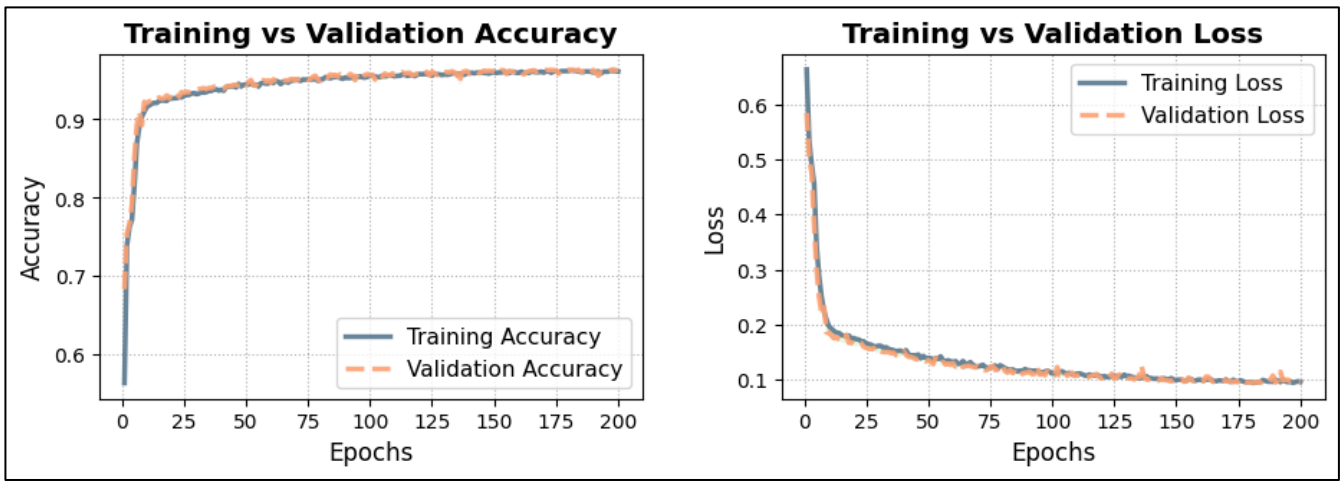

Figure 7. Training and validation results

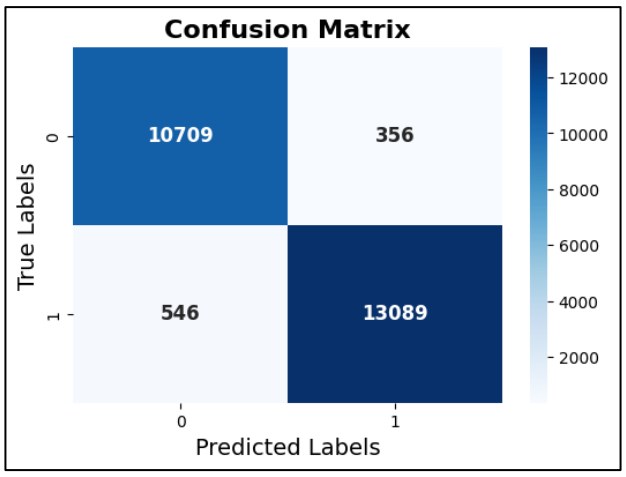

Figure 8. Confusion matrix

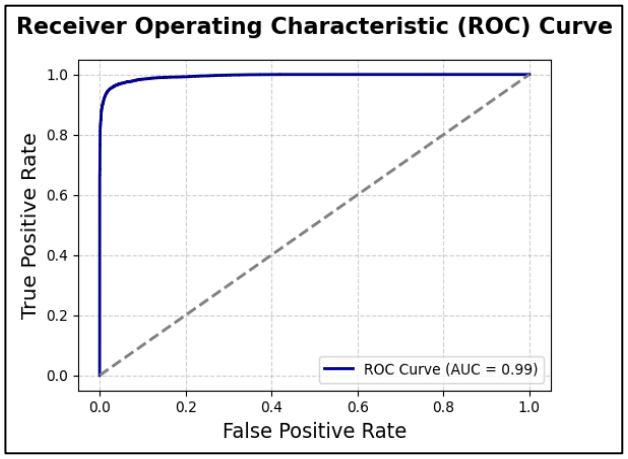

Figure 9. ROC curve

In a real-world intrusion detection scenario, a model with a higher AUC is highly effective at detecting both novel attacks and known patterns, decreasing the risk of effective intrusions and increasing total network security. This ROC curve proves that the model is highly reliable and prepared for deployment in network security systems.

The radar chart of various classification metrics is shown in Figure 10. A radar chart is beneficial for visualizing multiple performance metrics simultaneously, making it easy to identify the strengths and weaknesses of a model. From the chart, it is clear that the model achieves high values in F1-Score, Precision, Recall, Specificity, and Accuracy, denoting that the model is performing well in identifying both normal and malicious traffic. The values for Recall and F1-Score are 96.45% and 96.55%, indicating that the model is highly efficient at detecting attacks while diminishing false negatives. The specificity of 95.89% reflects that the model also avoids false positives, ensuring that normal traffic is not misclassified as malicious. The approach sustains stable performance along with distinct classes with an accuracy of 96.20%, although the distribution of data is unbalanced. The developed approach is assisted by a few enhancements in calibration, with the lesser values of Log loss and Cohen's kappa of 0.0976 and 0.9233. The Matthews Correlation Coefficient (MCC) is 0.9233, which denotes that the approach is proper and a requirement for fine-tuning to enhance correlation among true and predicted labels. It reveals that the approach is balanced, which is a good candidate for applications in network security.

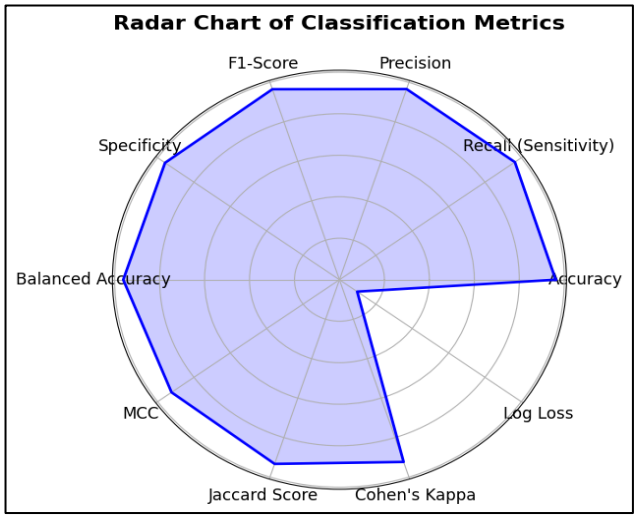

Figure 10. Radar chart of classification metrics

The developed approach is compared with MLP [21], Decision Tree (DT) [22], and RF [23], which are depicted in Figure 11. The accuracy of 96.2% is attained by the proposed research that integrates modern optimization in intrusion detection. It reveals that the developed approach effectively acquires the patterns of malicious activities, thereby improving the reliability of detection and diminishing the chances of false positives. The analysis of performance metrics for the developed approach with RF [24] and Support Vector Machine with Gaussian Mixture Model (SVM-GMM) [25] is revealed in Table 1. The developed approach has a recall of 96.45%, a precision of 96.64 %, and an F1 score of 96.55 %. It denotes that it acquires complicated patterns linked with network intrusions, sustains high accuracy, and diminishes false negatives and positives. Therefore, this research is more reliable and efficient for intrusion detection.

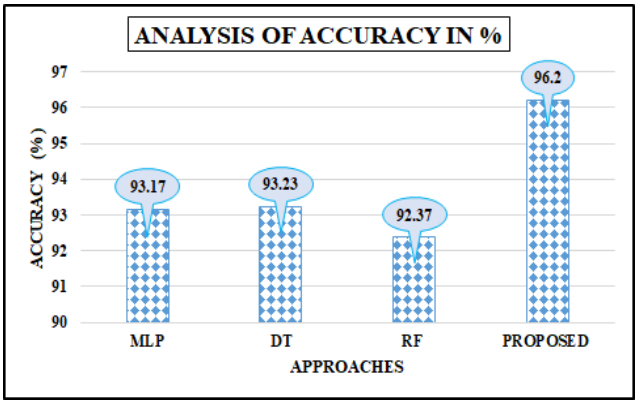

Figure 11. Comparative analysis of accuracy

Table 1. Comparison of evaluation metrics

APPROACHES	PRECISON (%)	RECALL (%)	F1-SCORE (%)
RF	94.80	95.70	95.10
SVM-GMM	96.10	95.65	95.85
PROPOSED	96.64	96.45	96.55

The comparison of the Mathews Correlation Coefficient (MCC) for the developed approach with the Bayesian [26] and RNN [27] approach is illustrated in Figure 12. The proposed approach has an MCC of 92.33%, which is better than other existing methods, denoting it delivers an accurate classification of network intrusions. This better value of MCC indicates the improved management of difficult intrusion patterns and diminished false rates, thereby enhancing the IDS's reliability.

The comparative analysis of ROC value for K Nearest Neighbors (KNN) [28] and SVM [26] classifiers with a developed approach is seen in Figure 13. The developed method has a higher ROC of 0.99 than other approaches, indicating its improved capability of classification. It enhances the efficacy of IDS by reducing the false negatives and positives. The higher ROC value of the developed approach highlights its ability to accurately distinguish between normal and attack traffic, making it a more reliable solution for IDS.

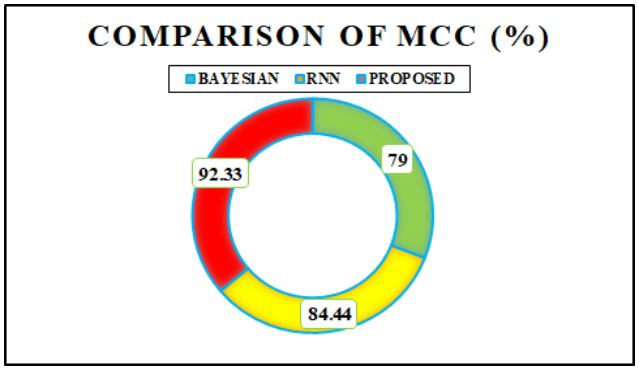

Figure 12. Analysis of MCC

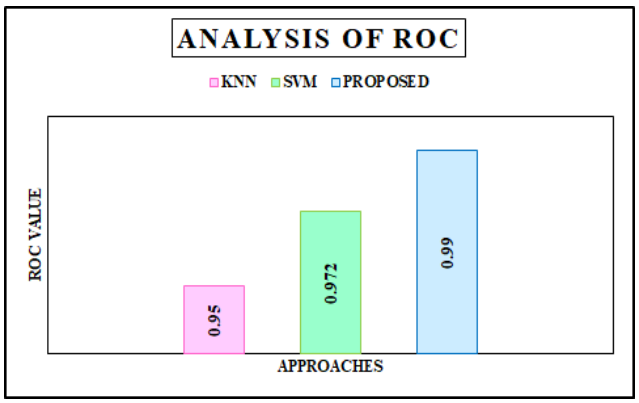

Figure 13. Analysis of ROC

4. Conclusion

This paper implements a classification approach for DL-based network intrusion detection systems. The preprocessing approach ensures that the overall quality of data is enhanced, which improves the accuracy of IDS. Then, the Exploratory Data Analysis detects hidden patterns, identifies anomalies, and recognizes relationships among network variables. By eliminating irrelevant or redundant features, RFE enhances the performance of DL models used in IDS, leads to more accurate recognition of intrusions, and decreases the computational cost. Subsequently, the MLPRNN classifier, leveraging the strengths of both MLP and RNN classifiers, attains higher accuracy and better performance in detecting and mitigating security threats and is optimized with the Zebra-Falcon finch optimization algorithm. The developed work is applied in Python software, and a comparison with conventional techniques reveals the performance of this research. According to experimental results, the developed approach outperforms certain recently described network intrusion detection techniques on the full dataset in terms of performance metrics with an accuracy of 96.20% and the capability to correctly detect various types of network intrusions.

Ethical issue

The authors are aware of and comply with best practices in publication ethics, specifically with regard to authorship (avoidance of guest authorship), dual submission, manipulation of figures, competing interests, and compliance with policies on research ethics. The authors adhere to

publication requirements that the submitted work is original and has not been published elsewhere.

Data availability statement
The manuscript contains all the data. However, more data will be available upon request from the authors.

Conflict of interest
The authors declare no potential conflict of interest.

References

[1] P. Barnard, N. Marchetti and L. A. DaSilva, "Robust network intrusion detection through explainable artificial intelligence (XAI)," IEEE Networking Letters, vol. 4, no. 3, pp.167-171, 2022. https://doi.org/10.1109/LNET.2022.3186589

[2] S. Mishra, "An optimized gradient boost decision tree using enhanced African buffalo optimization method for cyber security intrusion detection," Applied Sciences, vol. 12, no. 24, pp. 12591, 2022. https://doi.org/10.3390/app122412591

[3] R. Mohammad, F. Saeed, A. A. Almazroi, F. S. Alsubaei and A. A. Almazroi, "Enhancing Intrusion Detection Systems Using a Deep Learning and Data Augmentation Approach," Systems, vol. 12, no. 3, pp. 79, 2024. https://doi.org/10.3390/systems12030079

[4] S. A. Bakhsh, M. A. Khan, F. Ahmed, M. S. Alshehri, H. Ali and J. Ahmad, "Enhancing IoT network security through deep learning-powered Intrusion Detection System," Internet of Things, vol. 24, pp. 100936, 2023. https://doi.org/10.1016/j.iot.2023.100936

[5] S. More, M. Idrissi, H. Mahmoud and A. T. Asyhari, "Enhanced intrusion detection systems performance with UNSW-NB15 data analysis," Algorithms, vol. 17, no. 2, pp. 64, 2024. https://doi.org/10.3390/a17020064

[6] M. A. Khan, A. Rehman, K. M. Khan, M. A. Al Ghamdi and S. H. Almotiri "Enhance intrusion detection in computer networks based on deep extreme learning machine," Computers, Materials & Continua, vol. 66, no. 1, 2021. http://dx.doi.org/10.32604/cmc.2020.013121

[7] W. A. Ghanem, S. A. Ghaleb, A. Jantan, A. B. Nasser, S. A. Saleh, A. Ngah, A. C. Alhadi, H. Arshad, A. M. Saad, A. E. Omolara and Y. A. El-Ebiary, "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. https://doi.org/10.1109/ACCESS.2022.3192472

[8] M. Sajid, K. R. Malik, A. Almogren, T. S. Malik, A. H. Khan, J. Tanveer and A. U. Rehman, "Enhancing intrusion detection: a hybrid machine and deep learning approach," Journal of Cloud Computing, vol. 13, no. 1, pp. 123, 2024. https://doi.org/10.1186/s13677-024-00685-x

[9] J. M. Valente and S. Maldonado, "SVR-FFS: A novel forward feature selection approach for high-frequency time series forecasting using support vector regression," Expert Systems with Applications, vol. 160, pp. 113729, 2020. https://doi.org/10.1016/j.eswa.2020.113729

[10] S. Farahdiba, D. Kartini, R. A. Nugroho, R. Herteno and T. H. Saragih, "Backward elimination for feature selection on breast cancer classification using logistic regression and support vector machine algorithms," IJCCS (Indonesian Journal of Computing and Cybernetics Systems), vol. 17, no. 4, pp. 429-440, 2023. https://doi.org/10.22146/ijccs.88926

[11] C. Ioannou and V. Vassiliou, "Network attack classification in IoT using support vector machines," Journal of sensor and actuator networks, vol. 10, no. 3, pp. 58, 2021. https://doi.org/10.3390/jsan10030058

[12] N. Zhu, C. Zhu, L. Zhou, Y. Zhu and X. Zhang, "Optimization of the random forest hyperparameters for power industrial control systems intrusion detection using an improved grid search algorithm," Applied Sciences, vol. 12, no. 20, pp. 10456, 2022. https://doi.org/10.3390/app122010456

[13] M. Alazab, R. A. Khurma, P. A. Castillo, B. Abu-Salih, A. Martín and D. Camacho, "An effective networks intrusion detection approach based on hybrid Harris Hawks and multi-layer perceptron," Egyptian Informatics Journal, vol. 25, pp. 100423, 2024. https://doi.org/10.1016/j.eij.2023.100423

[14] S. Ho, S. Al Jufout, K. Dajani and M. Mozumdar, "A novel intrusion detection model for detecting known and innovative cyberattacks using convolutional neural network," IEEE Open Journal of the Computer Society, vol. 2, pp. 14-25, 2021. https://doi.org/10.1109/OJCS.2021.3050917

[15] S. M. Kasongo, "A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework," Computer Communications, vol. 199, pp. 113-125, 2023. https://doi.org/10.1016/j.comcom.2022.12.010

[16] V. Saravanan, M. Madiajagan, S. M. Rafee, P. Sanju, T. B. Rehman and B. Pattanaik, "IoT-based blockchain intrusion detection using optimized recurrent neural network," Multimedia Tools and Applications, vol. 83, no. 11, pp. 31505-31526, 2024. https://doi.org/10.1007/s11042-023-16662-6

[17] R. Chaganti, A. Mourade, V. Ravi, N. Vemprala, A. Dua and B. Bhushan, "A particle swarm optimization and deep learning approach for intrusion detection system in internet of medical things," Sustainability, vol. 14, no. 19, pp. 12828, 2022. https://doi.org/10.3390/su141912828

[18] M. Nkongolo, J. P. Van Deventer, S. M. Kasongo, S. R. Zahra and J. Kipongo, "A cloud based optimization method for zero-day threats detection using genetic algorithm and ensemble learning," Electronics, vol. 11, no. 11, pp. 1749, 2022. https://doi.org/10.3390/electronics11111749

[19] N. kumar Bukka, S. Jagadeesh and K. S. Reddy, "Autoencoder-based Deep Learning Approach for Intrusion Detection System using Firefly Optimization Algorithms," 2024. https://doi.org/10.21203/rs.3.rs-4076341/v1

[20] C. Ganguli, S. K. Shandilya, M. Gregus and O. Basystiuk, "Adaptive Network Sustainability and Defense Based on Artificial Bees Colony Optimization Algorithm for Nature Inspired Cyber Security," Computer Systems

Science & Engineering, vol. 48, no. 3, 2024. http://dx.doi.org/10.32604/csse.2024.042607
[21] M. Alazab, R. A. Khurma, P. A. Castillo, B. Abu-Salih, A. Martín and D. Camacho, "An effective networks intrusion detection approach based on hybrid Harris Hawks and multi-layer perceptron," Egyptian Informatics Journal, vol. 25, pp. 100423, 2024. https://doi.org/10.1016/j.eij.2023.100423
[22] M. Ahmad, Q. Riaz, M. Zeeshan, H. Tahir, S. A. Haider and M. S. Khan, "Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using UNSW-NB15 data-set," EURASIP Journal on Wireless Communications and Networking, vol. 2021, pp. 10, 2021. https://doi.org/10.1186/s13638-021-01893-8
[23] H.M. Alshahrani, "Coll-iot: A collaborative intruder detection system for internet of things devices," Electronics, vol. 10, pp. 848, 2021. https://doi.org/10.3390/electronics10070848
[24] H. A. Ahmed, A. Hameed and N. Z. Bawany, "Network intrusion detection using oversampling technique and machine learning algorithms," PeerJ Computer Science, vol. 8, pp. e820, 2022. https://doi.org/10.7717/peerj-cs.820
[25] C. Wang, Y. Sun, S. Lv, C. Wang, H. Liu and B. Wang, "Intrusion detection system based on one-class support vector machine and Gaussian mixture model," Electronics, vol. 12, no. 4, pp. 930, 2023. https://doi.org/10.3390/electronics12040930
[26] Y. S. Almutairi, B. Alhazmi and A. A. Munshi, "Network intrusion detection using machine learning techniques," Advances in Science and Technology Research Journal, vol. 16, no. 3, pp. 193-206, 2022. https://doi.org/10.12913/22998624/149934
[27] M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi and A. Razaque, "Deep recurrent neural network for IoT intrusion detection system, Simulation Modelling Practice and Theory," 101, 2020. https://doi.org/10.1016/j.simpat.2019.102031
[28] U. Ahmed, M. Nazir, A. Sarwar, T. Ali, E. H. Aggoune, T. Shahzad and M. A. Khan, "Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering," Scientific Reports, vol. 15, no. 1 pp. 1726, 2025. https://doi.org/10.1038/s41598-025-85866-7