**International Journal of Electrical and Computer Engineering (IJECE)**
Vol. 15, No. 6, December 2025, pp. 5570~5583
ISSN: 2088-8708, DOI: 10.11591/ijece.v15i6.pp5570-5583 

# Improving network security using deep learning for intrusion detection

**Mohammed Al-Shabi[1], Anmar Abuhamdah[1], Malek Alzaqebah[2,3]**
[1]Department of Management Information System, College of Business Administration, Taibah University, Al Madinah Al Munawwarah, Saudi Arabia
[2]Department of Mathematics, College of Science, Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia
[3]Basic and Applied Scientific Research Center, Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia

**Article Info**

***Article history:***

Received Nov 22, 2024
Revised Apr 12, 2025
Accepted May 24, 2025

***Keywords:***

Deep learning
Intrusion detection
Machine learning
Network security
Random forest

**ABSTRACT**

As cyber threats and network complexity grow, it is crucial to implement effective intrusion detection systems (IDS) to safeguard sensitive data and infrastructure. Traditional methods often struggle to identify sophisticated attacks, necessitating advanced approaches like machine learning (ML) and deep learning (DL). This study explores the application of ML and DL algorithms in IDS. Feature selection techniques, such as correlation and variance analysis, were employed to identify key factors contributing to accurate classification. Tools like WEKA and MATLAB supported data pre-processing and model development. Using the UNSW-NB15 and NSL-KDD datasets, the study highlights the superior performance of random forest (RF) and multi-layer perceptron (MLP) algorithms. RF ensemble decision trees and MLP multi-layered architecture enable accurate attack detection, demonstrating the potential of these advanced techniques for enhanced network security.



***Corresponding Author:***

Anmar Abuhamdah
Department of Management Information System, College of Business Administration, Taibah University
42353 Medina, al-Madinah al-Munawwarah, Kingdom of Saudi Arabia
Email: aabuhamdah@taibahu.edu.sa

## 1. INTRODUCTION

The rapid growth in data, technology, and smart devices has significantly increased reliance on internet-connected systems, which are now integral to various aspects of daily life [1]–[3]. As a result, ensuring the integration and preservation of data online has become a critical requirement for any system operating in an online environment. Network administrators face increasing challenges in maintaining security and shielding data against malware and cyberattacks [2], [3]. The ever-evolving variety and sophistication of these threats necessitate a continuous enhancement of security systems to ensure data confidentiality, integrity, and seamless accessibility.

Artificial intelligence has played a pivotal role in anomaly detection by employing classification techniques. Widely available datasets like UNSW-NB15, NSL-KDD, CICIDS2017, and KDD-Cup'99 (KDDCUP99) have significantly contributed to the evaluation of machine learning (ML) algorithms and advancement of a cybersecurity measures [4], [5]. Deep learning (DL) techniques often outperform shallow methods when aggregating several learning models. This method permits the discovery and exploitation of strengths in each model. Therefore, classification accuracy increases and complex attack patterns can be detected better [6]–[8].

The difficulty in intrusion detection lies in high-dimensional, imbalanced dataset, which reduces classification accuracy and results in high computational cost [9]–[11]. Moreover, traditional ML solutions

do not generalize well to diverse attack types, especially zero-day attacks [12], [13]. This paper aims to address these limitations by utilizing feature selection methods and hybrid ML-DL models to improve detection accuracy and decrease false positives.

Given the high volume of access attempts within networks, effective logging of activities is vital. This can be achieved through software, hardware, or a combination of both. Intrusion detection needs robust approaches, including the exploitation of constantly updated network services and databases, to stay aligned with the increasing diversity of attacks. This also requires knowledge in both intrusion and anomaly detection [14]. The progression of malicious methods highlights the necessity for security systems that not only support confidentiality, integrity, and accessibility but also guarantee operational continuity over accurate logging of network activities. This comprises consistent updates to network servers and databases and tailoring configurations to align with specific security objectives [15].

The primary goal of this study is to enhance the detection of intrusions in computer networks through the use of ML and DL-based algorithms. Precisely, we seek to optimize characteristic selection, contrast the behavior of ML and DL algorithms and propose an efficient model that detects the different categories of network attacks.

Designing an intrusion detection system classically involves four necessary steps [4]. These include:

a. Data collection by gathering detailed network traffic information, including traffic type, host, protocol, and other relevant details.
b. Feature extraction by filtering the collected data to retain only the most relevant features for analysis.
c. Data analysis to evaluate the selected features to determine whether the data represents normal traffic or a potential attack.
d. Action implementation that takes appropriate measures based on the analysis results, such as issuing alerts to administrators or mitigating threats by blocking network ports or halting operations temporarily.
e. Feature selection and extraction are crucial steps in data cleansing for intrusion detection systems (IDS). Approaches such as ML-based filtering or ensemble learning are commonly employed. Ensemble learning, which collections predictions from several algorithms, has recognized effective in improving accuracy and accomplishing better results [15].

This research adds to the literature by assessing and comparing the performance of traditional ML versus DL for intrusion detection. We use correlation based and variance feature selection mechanisms to improve classification accuracy. Second, we evaluate our solutions on two publicly known datasets, UNSW-NB15 and NSL-KDD, and show that DL based models, notably MLP and random forest (RF) outstrip their traditional counterparts from detection accuracy and F1-score perspectives.

The focus of this study is to improve security breach detection in computer networks with ML and DL algorithms. In particular, we want to assess their effectiveness on the UNSW-NB15 and NSL-KDD datasets, so that their detection rate is maximized and the computational overhead is also minimized. These initiatives illustrate how security issues are changing over time. ML and DL are an integral part in combating these threats. Learning from new data allows them to react to and detect new and advanced cyber-attacks even if they are merely nominal tools in doing so. The remainder of the paper is organized as: section 2 presents related works, section 3 describes the method, section 4 presents the results and discussion, and section 5, finally we conclude with the discussion and the future work.

## 2. RELATED WORKS

The growing density and complexity of network attacks, combined with the growing capabilities of cybercriminals, have made securing networks a critical importance for organizations. Effective disruptions of networks and websites by hackers highlight the persistent need for reliable IDS. These progressions in hacking methods have further underlined the implication of IDS in modern cybersecurity [16]–[18].

This study contributes to the field by evaluating and comparing the performance of traditional ML algorithms against DL methods for intrusion detection. It focuses on how these models perform on standard datasets such as UNSW-NB15 and NSL-KDD. The findings provide insights into the effectiveness of different approaches in detecting network intrusions. It also observes how accuracy can be enhanced by exploiting advanced feature extraction methods. To provide a complete comparison, the research uses two noticeable datasets, concentrating on preprocessing processes that show an essential role in improving detection accuracy.

Numerous studies have explored the use of artificial intelligence-based approaches for IDS, particularly leveraging ML and DL techniques. We present an up-to-date structured review of recent papers, addressing the analysis of methodology, results, strengths, and limitations.

a. A study in [6] offered a hybrid learning method that was parametric and non-parametric classifier combination for IDS. This method achieved a considerable improvement of detection accuracy and ability of reducing false positives, particularly on unbalanced corpus as UNSW-NB15.

b. One more work in [7], preferred ensemble based DL models for IoT IDS. However, by combining CNNs and RNNs the system had increased precision and recall, but it was computationally expensive.
c. In [8], presented a systematic literature review on the DL techniques for cyber threat detection in IoT networks. It drew attention to the value of feature extraction and temporal modeling, and indicated circumstances in which DL can surpass classical ML.
d. In [9], also suggested improving IDS using DL and data augmentation techniques. Their approach enhanced the detection of minority classes albeit at the cost of model complexity.
e. Comparison of binary and multi-class classification performance of ML & DL models was studied in [10]. The results showed that deep models, such as MLP and LSTM, obtained better performance (accuracy and F1-score) than conventional classifiers.
f. Reference [11] performed in-depth analysis of ML techniques for class imbalance in IDS. Methods including SMOTE and cost-sensitive learning were indicated to improve accuracy on under-represented attack types.
g. The unsupervised learning for the detection of the zero-day attacks is also proposed in [12]. The model was effective in detecting anomalies, however operational problems were due to the fact that it was required to tune thresholds.
h. A critical review in [13], presented AI-based IDS solutions, highlighting the importance of the adaptive and scalable solutions. It pointed out that the DL models are very powerful, but they are data hungry and big labeled datasets do not exist.
i. In [18], the ANNs were used to classify network intrusion on the NSL-KDD dataset. Their model was able to detect HDP with 95% accuracy. This approach worked reasonably well but had difficulty with high dimensional features and generalization to new types of attack.
j. Reference [19] presented a two-stage model based on decision tree algorithms for network intrusion detection although the CICIDS2017 achieved a classification accuracy of 92.6% that was better than the existing methods. But it was computationally heavy and was not scalable in real-time applications.
k. Reference [20] presented an analysis of several ML-based IDS models, specializing in decision trees (DT) with the UGR'16 dataset. DT achieved more than 94% accuracy for correctly classifying new content, implying resilience to known attacks at the cost of its inability to recognize zero-day attacks.
l. Reference [21] proposed the DEHO model for the detection distributed denial-of-service (DDoS) attacks in cloud. DEHO achieved better accuracy than other classifiers on four datasets. Its hybrid architecture was beneficial to the detection but demanded a long training time.
m. In [22], designed a stacked LSTM with autoencoder encryption to enhance DL functionality in intrusion detection. Although the model decreased the amount of errors in predicting, and made traffic decisions more accurate, it lacked interpretability, and cost too much to deploy.
n. In study [23], presents the UNSW-NB15 dataset which has also emerged as a popular target for benchmarking network. One of the most important benefits of this dataset is that it covers a variety of modern attack types from real network traffic by up-to-date tools and provides a more realistic environment than older datasets such as KDD99. It's also packed with well-designed capabilities to make training and evaluating machine learning models easy. But one of the most significant flaws is the class imbalance in the dataset, which might bias learning algorithms and decrease the detection performance regarding minority attack types. Furthermore, the dataset has been generated in 2015 and therefore may not adequately represent the most recent threat landscapes and more advanced cyber-attacks occurring in more recent years.
o. In reference [24], the KDDCUP99, the NSL-KDD, and UNSW-NB15 datasets were analyzed in internet of things (IoT) contexts using DL. 3. It also highlighted that the more recent datasets, for example, UNSW-NB15 have more valid evaluations than for example KDDCUP99.
p. Authors in [25] proposed a guidelines for training based on KDD-Cup'99 and NSL-KDD dataset for training of anomaly-based IDS. While valuable, these datasets are now believed less relevant to the way modern attacks operate.

The literature survey shows a clear migration toward the deployment of DL-based IDS especially with the use of both hybrid architecture and ensemble methods. These models are generally better than classical ML algorithms in the detection accuracy, precision and F1-score specifically when trained with recent datasets as UNSW-NB15. But there are still some limitations such as huge computational cost, no interpretation and poor effects for imbalanced data. The recent literature highlights that data augmentation, feature selection and unsupervised learning are crucial to dealing with these problems. Future work will look into coping with real-time adaptation, lightweight DL models, and cross-dataset generalization in an effort to achieve a complete adaptability against incoming threats.

## 3. METHOD

The efficiency of ML and DL algorithms for intrusion detection is investigated in this paper with two well-known benchmark datasets: UNSW-NB15 and NSL-KDD [23], [24], both usually applied in network security research under controlled experimental conditions. These datasets capture a diversity of network activities and attack types. The research applies to a combination of ML models and DL techniques, such as RF and MLP. To enhance classification performance, the datasets underwent preprocessing, and feature selection techniques were employed to identify the most significant attributes.

UNSW-NB15 dataset is a benchmark commonly used in cybersecurity research for assessing IDS. Founded at the Australian Center for Cyber Security in 2015 to overcome the limitations of legacy datasets including NSL-KDD and KDDCUP99, mainly in expressive modern network behaviors and diverse attack scenarios [23]–[26]. The dataset simulates realistic network traffic, blending normal activities and numerous modern attack patterns, to evaluate IDS models effectively. Where the dataset structure as:

The dataset contains 49 features representing various network attributes such as IP addresses, ports, protocols, packet sizes, and timestamps.

It comprises approximately 2.5 million records, divided into normal traffic and attack traffic.

It includes a range of attack categories that reflect real-world scenarios, such as:

a. Fuzzers: sending random inputs to discover vulnerabilities.
b. Analysis: port scanning and other probing techniques.
c. Backdoors: unauthorized access through covert channels.
d. Denial of service (DoS): overwhelming a network or server with traffic.
e. Exploits: using software vulnerabilities to compromise systems.
f. Generic: platform-agnostic attacks, like password cracking.
g. Reconnaissance: gathering information for potential future attacks.
h. Shellcode: malicious code designed to exploit vulnerabilities in software to execute unauthorized commands on a target system.
i. Worms: self-replicating malware spreading across networks.

UNSW-NB15 key attributes are the attack-cat: specifies the type of attack in the dataset (e.g., DoS, Exploit), label: specifies whether the record is normal (0) or an attack (1), and distribution: the dataset has an almost equal balance between normal and attack traffic to reduce bias in classification.

Figure 1 presents the dataset, which consists of 2,540,044 records along with two classification tools: attack-cat and label. The 'delay' attribute indicates the presence of a defect, with a value of 0 for normal records and 1 for defective records. The general classification can be performed using either or both of the classification tools, leading to identical classification results. In our research, the UNSW-NB15 attribute is identified as the most important feature.

| UNSW-NB15 Record Type | Number of Records | Percentage of Records |
|---|---|---|
| Normal | 2,218,761 | 87.36% |
| Fuzzers | 24246 | 0.95% |
| Analysis | 2677 | 0.11% |
| Backdoor | 2329 | 0.09% |
| Dos | 16353 | 0.64% |
| Exploits | 44525 | 1.75% |
| Generic | 215481 | 8.48% |
| Reconnaissance | 13987 | 0.55% |
| Shellcode | 1511 | 0.06% |
| Worms | 174 | 0.07% |

Figure 1. Percentage distribution of records in the UNSW-NB15 dataset

A detailed overview of the dataset's features is provided in Figure 2. NSL-KDD dataset is a widely used resource for evaluating IDS. It is derived from the KDDCUP99 dataset and addresses several issues present in the original version by removing redundant records [1], [25]. This modification results in a more balanced dataset and minimizes bias that classifiers might otherwise develop.

The NSL-KDD dataset contains 41 attributes, along with a class attribute that categorizes the type of connection between two records (i.e. Normal and Abnormal (attack) traffic). Table 1 illustrates the percentage distribution of records in this database (i.e. # indicate the number of records, where % indicate the percentage of records overall). It includes both normal and abnormal (attack) records, which are classified into five categories [24]–[26]:

a. normal traffic.
b. Dos: attempts to make the device unavailable to its users.
c. Probe: efforts to gather information to identify potential vulnerabilities.
d. User 2 root (U2R): unauthorized access to the system is obtained.
e. Remote to local (R2L): the attacker gains unauthorized access from a remote machine.

The NSL-KDD dataset consists of 41 features and one class attribute which characterizes each instance as normal/ one of various attack types as shown in Figure 3. These features are grouped into three categories: basic features, Content features, and Traffic features (for more details please read [24], [25]). Feature extraction is a predictive technique that reduces the number of input variables during model development. By eliminating irrelevant or redundant features, this process simplifies the model architecture. As a result, computational efficiency improves, and the model becomes more effective at identifying meaningful patterns in the data. It is especially beneficial when working with ML models that process large datasets. Feature extraction helps identify the most significant features that are strongly correlated with the target variable [23], [27].

| Feature Name | | | | | | |
|---|---|---|---|---|---|---|
| **(1-7)** | **(8-14)** | **(15-21)** | **(22-28)** | **(29-35)** | **(36-42)** | **(42-49)** |
| Srcip | Sbytes | Sload | dtcpb | Stime | is_sm_ips_ports | ct_dst_ltm |
| Sport | Dbytes | Dload | smeansz | Ltime | ct_state_ttl | ct_src_ltm |
| Dstip | Sttl | Spkts | dmeansz | Sintpkt | ct_flw_http_mthd | ct_src_dport_ltm |
| Dsport | Dttl | Dpkts | trans_depth | Dintpkt | is_ftp_login | ct_dst_sport_ltm |
| Proto | Sloss | swin | res_bdy_len | tcprtt | ct_ftp_cmd | ct_dst_src_ltm |
| State | dloss | dwin | Sjit | synack | ct_srv_src | attack_cat |
| Dur | service | stcpb | Djit | ackdat | ct_srv_dst | Label |

Figure 2. Features of the UNSW-NB15 dataset

Table 1. Percentage distribution of records in the NSL-KDD dataset

| NSL-KDD Record Type | KDDTrain+ (Training Set) | | KDDTest+ (Testing Set) | | Combined | |
|---|---|---|---|---|---|---|
| | # | % | # | % | # | % |
| Normal | 67,343 | 53.46 | 9,711 | 43.08 | 77,054 | 51.88 |
| Dos | 45,927 | 36.45 | 8,456 | 37.01 | 54,383 | 36.62 |
| Probe | 11,656 | 9.25 | 873 | 3.87% | 12,529 | 8.44 |
| R2L | 995 | 0.79 | 3,249 | 15.02 | 4,244 | 2.86 |
| U2R | 52 | 0.04 | 255 | 1.02 | 307 | 0.21 |

| Feature Name | | | | | |
|---|---|---|---|---|---|
| **(1-7)** | **(8-14)** | **(15-21)** | **(22-28)** | **(29-35)** | **(36-42)** |
| Duration | Wrong fragment | Su attempted | Is guest login | Same srv rate | Dst host same src port rate |
| Protocol type | Urgent | Num root | Count | Diff srv rate | Dst host srv diff host rate |
| Service | Hot | Num file creations | Srv count | Srv diff host rate | Dst host serror rate |
| Flag | Number failed logins | Num shells | Serror rate | Dst host count | Dst host srv serror rate |
| Source bytes | Logged in | Num access files | Srv serror rate | Dst host srv count | Dst host rerror rate |
| Destination bytes | Num compromised | Num outbound cmds | Rerror rate | Dst host same srv rate | Dst host srv rerror rate |
| Land | Root shell | Is host login | Srv rerror rate | Dst host diff srv rate | Class label |

Figure 3. Features of the NSL-KDD dataset

In this research, correlation, coefficients and variance are used to select the most relevant features for each dataset. Variance quantifies the extent to which feature values deviate from the mean. It serves as an indicator of how much the values differ from one another and from the average. A high variance indicates that the values are spread across a broader range, whereas a low variance suggests that the values are clustered near the mean. The formula for variance is given by (1).

$$COX(x) = 1/n \sum_{i=0}^{n} (xi - x) \quad (1)$$

where: $x$ is the arithmetic mean of $n$ values, $Xi$ is the number of samples.

In this research, variance is employed to determine which features show the greatest spread (variation) in their values. Features with higher variance are generally more effective in differentiating

between various classes. The coefficient of correlation measures the degree of linear association between two or more variables and aids prediction of one variable on the basis of another. When two variables are strongly correlated, knowing one can help predict the other. If the two variables are identical, only one parameter is needed to represent both, as the second parameter does not provide additional information. The correlation coefficient is defined by the following (2).

$$Corr(x,y) = \frac{Cov(x,y)}{ax.ay} \quad (2)$$

where, *Cov(x,y)* is the covariance between *x* and *y*, *ax* represents the standard deviation of *x*, and *ay* is the standard deviation of *y*.

In this study, the coefficient of correlation is utilized to estimate the impact magnitude of features on the target variable. Features with a higher absolute correlation coefficient (closer to 1 or -1) are more likely to be valuable for classification purposes. In this research, the adjustment and correlation coefficient methods were practical to the NSL-KDD and UNSW-NB15 datasets to classify the most important features linked to the target classes. This process is vital for improving classification accuracy and decreasing the computational load throughout the ML process.

The following steps were followed in the feature extraction process:

a. The variance for each feature was computed to assess its distribution across the dataset.
b. The correlation coefficient was calculated between each feature and the target variable to evaluate the linear relationship.
c. Based on the calculated variance and correlation values, the features most relevant to the classification task were selected.

Artificial intelligence (AI) is a field of computer science motivated on emerging techniques that allow machines to achieve tasks classically requiring human intelligence. ML, a key factor of AI, enables computer systems to learn straight from data, examples, and experiences. Using programmed algorithms, ML analyzes input data to predict output values, continuously improving and optimizing its processes to enhance performance and develop intelligence over time [28]. This study applies both traditional ML and DL algorithms for classifying network intrusions. Some of the algorithms used include:

a. K-nearest neighbors (KNN), a simple algorithm to classify a dataset based on the classification of its nearest neighbors [5].
b. RF, a classification algorithm that creates a collection of DT and uses majority voting for the final prediction [29].
c. DT, a decision tree is a flow chart-like tree structure, where each node stands for a feature, and each leaf represents a class label [30].
d. Naïve Bayes (NB), a probabilistic classifier based on Bayes' theorem [31].
e. MLP, a DL model composed of several layers of neurons: input, hidden, and output layers where each neuron in these layers is applied activation function to capture complex patterns [32].
f. Long short-term memory (LSTM), a type of recurrent neural network (RNN) that is well-suited to learning from sequences of data, due to its ability to retain information over sequences [33].

Measuring classification accuracy alone is insufficient to assess the performance of each class individually [34]. Therefore, the confusion matrix index will be computed for the dataset using the correlation matrix as in Figure 4.

| Confusion Matrix | Predicted No | Predicted Yes |
|---|---|---|
| Actual No | TN (True Negative) | FP (False Positive) |
| Actual Yes | FN (False Negative) | TP (True Positive) |

Figure 4. Correlation matrix

The correlation matrix is a tool used to evaluate classifications by assessing their ability to correctly categorize samples, i.e., determining the correct class to which a sample belongs. When comparing predicted classifications with actual classifications, four possible outcomes can occur:

a. True positive (TP), where both the actual and predicted classifications are positive.
b. False positive (FP), real class is negative, but predicted is positive.
c. False negative (FN), the object's classification is positive while the predicted is less than 0.5.
d. True negative (TN), where both the actual and predicted classifications are negative.

By analyzing these outcomes, the following performance metrics are calculated:

a. Accuracy, that commonly used measure of overall classifier performance, calculated using (3).

$$Accuracy = \frac{TN+TP}{TN+TP+FN+FP} \quad (3)$$

b. Recall measures how well the model is able to find positive samples. It is the number of true positive predictions divided by the total number of actual positives. The recall is calculated as in (4).

$$recall = \frac{TP}{TotalActualYes} \quad (4)$$

c. Precision is a statistic that measures the fraction of the model's positive predictions that are correct. It refers to the number of true positive cases divided by the sum of true positive and false positive cases. The precision is computed using (5).

$$precision = \frac{TP}{TotalpredicatedYes} \quad (5)$$

d. F1 Score is the metrics that used to assess the performance of a classification model, particularly in situations with imbalanced class distributions. It takes into account both false positive and false negative errors, offering a balanced assessment. The F1 Score is calculated using the following (6).

$$F1 - Score = \frac{2*prec*recall}{prec+recall} \quad (6)$$

For this analysis, the data was split into training and testing data 80:20 % to provide sufficient data for training models, but also to allow a robust evaluation phase. Furthermore, 10-fold cross validation was adopted in training models to further improve model generalization ability and to minimize the over fitting. This method requires each example in the dataset to be used to both train and validate the model multiple times; hence, giving a more reliable estimate of model performance in the presence of various types of attacks. The hyper-parameters of all models were grid searched over the training set, and the final testing was performed on the test set. All results were averaged over three independent experiments with three different random seeds for checking consistency of results.

Figure 5 demonstrates a flowchart exactness the methodology used in this research. The process begins with data cleaning, which is vital before applying any algorithms to precise database anomalies. This includes addressing lost values based on their nature and adapting textual data into numerical format to decrease computational weight. Data normalization, and significant step scales the feature values to a range of [0, 1], further dropping computational demands. The UNSW-NB15 and NSL-KDD datasets were originally exported into Excel for organized visualization, basic analysis, and preprocessing. Missing values and duplicates were controlled to prevent bias or negative impacts on the classifier. In Excel, attack types were encoded into numerical values to facilitate easier classification with selected classifiers.

Feature extraction, a serious step in building an AI model, reduces the dimensionality of high-dimensional data, alleviating the computational burden and simplifying the model. This step is dynamic since increased complexity can negatively affect training and testing times, thus impacting the accuracy of intrusion detection. Feature selection, based on correlation coefficients, was used to identify the most relevant features for the task. These steps constitute the preliminary data preprocessing phase, which is crucial for reducing processing and classification time while enhancing classification accuracy. Irrelevant features can often have a detrimental effect on classification algorithms.

After feature selection, ML algorithms, specifically the MLP and LSTM networks, were applied for classification. The results were then compared with those from previous studies using assessment metrics for both datasets. The classifiers were adjusted with specific parameters, which might be attuned depending on the results and comparisons.

Hyperparameters of the machine learning and deep learning models were adjusted to maximize the performance through a carefully designed tuning process. Grid search with cross-validation (GridSearchCV) was used to search over a broad set of hyperparameters values for each method. Among the hyperparameters to be tuned were number of trees and maximum depth for RF model, regularization and learning rate for logistic regression, number of hidden layers, activation function and learning rate for MLP model, and sequence length and memory units for LSTM model. The best settings were selected by validation accuracy in 10-fold cross-validation. Early stopping was also used in training to avoid overfitting and computation cost. This fine-tuning strategy guaranteed all the models underwent the fine-tuning in full

to achieve best detection performance and generalization ability. The hyperparameters experimented for each algorithm are:

a. RF: Number of trees=120, learning rate=0.01
b. Logistic regression: Number of iterations=100, maximum depth of decision tree=default
c. LSTM: Activation function=tanh

The 10-fold cross-validation was used to determine the optimal performing configurations according to validation accuracy. Finally, early stopping was also used during training to avoid overfitting and to save computation burden. The structured tuning procedure was designed to obtain the best detection performance and generalization ability for all models. All the experiments were run on: MATLAB R2021a and WEKA 3.9 (PHE-RS) in an Intel Core I7 Processor and RAM 16 GB, where the average CPU usage varied on a range of 60–85%, and the memory was approximately 3.5 and 4.2 GB (according to the model). Windows 11 Pro was the operating system.

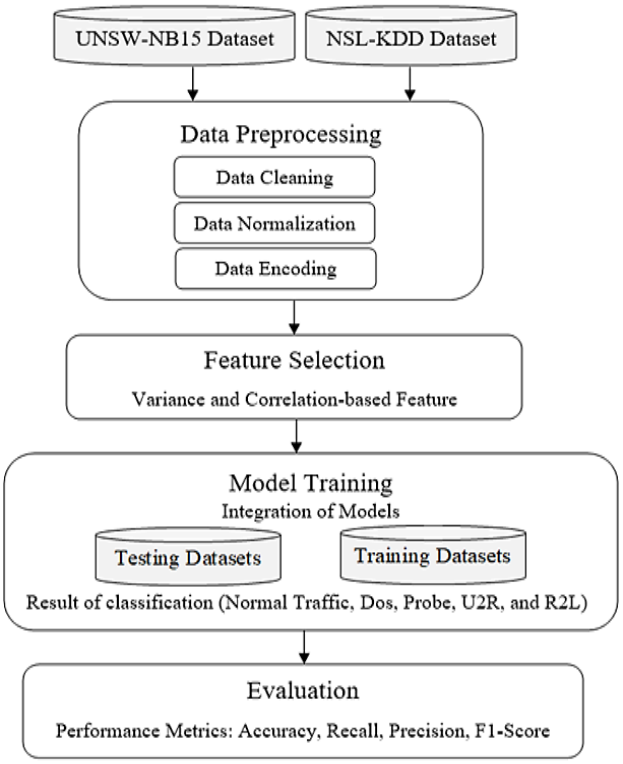


Figure 5. Workflow box diagram

## 4. RESULTS AND DISCUSSION

This section delivers an outline of the research methodology and the results obtained. The research compared the intrusion detection accuracy of the NSL-KDD and UNSW-NB15 datasets using ML and DL algorithms. Weka and MATLAB software were employed at numerous steps of the classification and feature extraction processes. The performance is measured first on a Naïve Bayes tool as a baseline algorithm, because of its annotation simplicity and runtime. We compare later models against this base model to evaluate how it improves detection in terms of accuracy, precision, recall, F1-score. Data preprocessing in Excel played an essential role in standardizing the data format. This step was essential for reducing processing and classification time, while also improving accuracy, as irrelevant features can negatively affect performance. The preprocessing phase involved:

a. Numerical encoding of attack types that converting attack types into numerical codes facilitated accurate classification. In this study, attacks were encoded in two stages: first, the main attack types were encoded, followed by the sub-attack types for more detailed identification. Normal records were encoded as 0 and attack records as 1 in both datasets. The study focused on binary classification.
b. Feature reduction can increase computational time and reduce classification accuracy, making feature reduction necessary. This research used mathematical relationships to calculate feature correlations and identify unnecessary features. This process included:

- Identifying redundant features via variance calculation, where features with zero variance, meaning those that remained unchanged across the dataset, were discarded.
- This was done using the variance formula in MATLAB with the variable function, where value represents the selected feature column. The variance results for the UNSW-NB15 dataset are presented in Table 2.

The reason for us to choose the ML and DL models, namely KNN, RF, MLP, LSTM, is due to the following several considerations specific to intrusion detection problems:

a. RF was selected due to its strength in dealing with high-dimensional data and overfitting avoidance and reasonable performance in multiclass classification tasks. It proves to be especially useful in the areas of cybersecurity, where the analysis of the importance of features is also a requirement.
b. MLP was chosen as a baseline deep learning model that can be applied effectively on to structured records such as network traffic logs. It is flexible in tuning the hidden layers and activation functions, so it can accommodate complex relations among features.
c. KNN was added as a simple non-parametric model that makes a solid baseline for the comparison with more sophisticated classifiers.
d. LSTM networks were added to test if the temporal aspects of network traffic namely time sequential attack behaviors if captured as a trainer feature or not by the ML models, and if the presence of such by the ML Models improves the pre- diction of attacks or not.

Together, these models constitute the most comprehensive comparison of classical machine learning and recent deep learning approaches in the context of IDS to date.

Table 2. Variance results for each feature in the UNSW-NB15 dataset

| FE. | Var. | FE. | Var. | FE. | Var. | FE. | Var. | FE. | Var. |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.3596 | 2 | 1.5859 | 3 | 0.2516 | 4 | 4.357 | 5 | 0.5839 |
| 6 | 0.02548 | 7 | 0.6987 | 8 | 0.0808 | 9 | 0.0171 | 10 | 0.0968 |
| 11 | 0.1202 | 12 | 0.8087 | 13 | 0.1322 | 14 | 5.205 | 15 | 1.5889 |
| 16 | 5.2958 | 17 | 5.3695 | 18 | 0.2547 | 19 | 0.2598 | 20 | 0.0256 |
| 21 | 0.2589 | 22 | 1.0259 | 23 | 0.0897 | 24 | 2.3698 | 25 | 8.3651 |
| 26 | 0.2203 | 27 | 0.5259 | 28 | 0.0023 | 29 | 0 | 30 | 0 |
| 31 | 0.0365 | 32 | 0.0027 | 33 | 5.2332 | 34 | 2.3658 | 35 | 0.3258 |
| 36 | 0.2581 | 37 | 0.6523 | 38 | 0.0968 | 39 | 0 | 40 | 0.0702 |
| 41 | 1.0056 | 42 | 0.1658 | 43 | 0.2365 | 44 | 0.8789 | 45 | 0.00364 |

Table 2 shows that features 29, 30, and 39 have zero variance, meaning their values remain constant across the entire dataset and are the same for all records. As a result, these features can be removed from the classification process, helping to reduce computational load and classification time. Feature 29 represents the start time, feature 30 represents the end time, and feature 39 indicates whether the user is logged in (with a value of 1) or not (with a value of 0). After eliminating these features, the dataset is reduced, leaving 44 features, along with the label indicating whether a record is normal or an attack. For the NSL-KDD dataset, the variance calculation results are shown in Table 3.

Table 3. Variance results for each feature in the NSL-KDD dataset

| FE. | Var. | FE. | Var. | FE. | Var. | FE. | Var. | FE. | Var. |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.98 | 5 | 2.235 | 6 | 4.5026 | 7 | 3.1042 | 8 | 0.0203 |
| 9 | 0.0013 | 10 | 0.862 | 11 | 0.0226 | 12 | 0.2467 | 13 | 52.847 |
| 14 | 0.0024 | 15 | 4.4353 | 16 | 64.6676 | 17 | 0.4581 | 18 | 0.0023 |
| 19 | 0.0046 | 20 | 0 | 21 | 0 | 22 | 0.0276 | 23 | 1.6522 |
| 24 | 7.9321 | 25 | 0.0872 | 26 | 0.089 | 27 | 0.1732 | 28 | 0.1732 |
| 29 | 0.1702 | 30 | 0.0672 | 31 | 0.0643 | 32 | 8.84296 | 33 | 1.2496 |
| 34 | 0.1898 | 35 | 0.0487 | 36 | 0.938 | 37 | 0.0073 | 38 | 0.0746 |
| 39 | 0.0794 | 40 | 0.1499 | 41 | 0.1607 | | | | |

Table 3 indicates that features 20 and 21 have zero variance, meaning their values remain constant across all records in the dataset. As a result, these features can be removed without impacting the classification accuracy, thus reducing the dataset size. After this step, the number of features is reduced to 39 (where feature 2 is the scr_port, feature 3 is the dst_ip and feature 4 is the dst_port). Feature 20 represents the number of commands issued in an FTP session, while feature 21 takes a value of 1 if a login to the hotlist occurs and 0 otherwise. Since both features have the same value for all records, removing them will not affect the classification process.

To identify correlated features, the correlation coefficient was calculated for each pair. The correlation coefficient ranges from -1 to 1, with values closer to 1 indicating a stronger correlation. The aim is to remove one feature from each highly correlated pair, as the presence of one feature makes the other redundant. Using the correlation coefficient formula and the MATLAB function corrcoef(a,b), the correlation coefficient for each pair was calculated. The following Table 4 presents the most correlated features, those with a correlation coefficient greater than 0.9, for the UNSW-NB15 and NSL-KDD datasets.

Table 4 reveals several correlations between features for UNSW-NB15 dataset: feature 10 is correlated with feature 31, feature 11 with feature 32, feature 45 with features 44 and 42, feature 46 with features 43, 41, and 47, feature 8 with features 17 and 23, and feature 9 with features 18 and 24. As a result, one feature from each correlated pair can be removed. For example, the number of connections with the same source IP and destination service, feature 45, can be substituted for features 44 (number of connections with the same source IP) and 42 (number of connections with the same destination service) since the last two features are auxiliary for the former. The same reason also applies to any other correlated feature. Following this process, the UNSW-NB15 dataset is reduced to 33 features, along with the label indicating whether a record is normal or abnormal.

Table 4. Correlated features in the UNSW-NB15and NSL-KDD datasets

| UNSW-NB15 | | NSL-KDD | |
|---|---|---|---|
| Feature pair | Correlation coefficient | Feature pair | Correlation coefficient |
| (10,31) | 0.9015 | (13,16) | 0.996 |
| (11,32) | 0.9253 | (12,16) | 0.9006 |
| (8,17) | 0.9906 | (25,26) | 0.9664 |
| (8,23) | 0.9915 | (25,38) | 0.9041 |
| (9,18) | 0.9934 | (25,39) | 0.9008 |
| (9,24) | 0.9959 | (26,38) | 0.9997 |
| (44,45) | 0.9742 | (26,39) | 0.921 |
| (42,45) | 0.9698 | (27,28) | 0.9755 |
| (41,46) | 0.9936 | (27,40) | 0.9852 |
| (43,46) | 0.9956 | (27,41) | 0.931 |
| (46,47) | 0.9041 | (28,40) | 0.98 |
| | | (28,41) | 0.9478 |
| | | (33,34) | 0.9044 |
| | | (38,39) | 0.945 |
| | | (40,41) | 0.9047 |

Table 4 also reveals several correlations between features for NSL-KDD dataset, where feature 16 is highly correlated with features 12 and 13, allowing features 12 and 13 to be removed while retaining feature 16. Similarly, feature 25 can replace features 26, 38, and 39; feature 28 can replace features 27, 40, and 41; and feature 33 can replace feature 34. For instance, feature 25 represents the percentage of retransmission attempts, which strongly correlates with feature 26, representing the percentage of connections to the same service and host. Since knowing the retransmission attempts provide sufficient information to infer successful connections, feature 26 becomes redundant. The same principle applies to the other correlated features. As a result, the retained features are 16, 25, 28, and 33, while the removed features are 12, 13, 26, 38, 39, 27, 40, and 41. This feature selection process reduces the dataset to 30 features.

Following this preprocessing step, classification algorithms were applied to both datasets, with the results compared using the Weka software. The following Table 5 presents the accuracy results based on confusion matrix parameters for each dataset for the UNSW-NB15 and NSL-KDD datasets. Table 5 demonstrates that all the classification algorithms produced strong performance, with the RF technique showing superior results on the UNSW-NB15 dataset, achieving the highest detection accuracy of 99.7% and an F1-score of 99%. Table 5 also shows that the DL model MLP achieved the highest performance on the NSL-KDD test set, with a detection accuracy of 98.9% and an F1-score of 99%. Overall, all classification algorithms delivered strong results, with performance metrics exceeding 90%. The following diagrams illustrate a comparison of performance metrics for both datasets.

In general, we observe that both ML and DL based models were promising in detecting network intrusions. Nonetheless, RF shows the highest performance on the UNSW-NB15 dataset (99.7%), while MLP achieved the same result on NSL-KDD dataset (98.9%). The result indicates that model performance may differ based on dataset properties, e.g. feature distribution and attack diversity. The following Figure 6 illustrates a comparison between algorithms on the x-axis (i.e. NB, KNN, RF, DT, MLP, and LSTM) of performance metrics in y-axis are divided into four sections (accuracy in Figure 6(a), recall in Figure 6(b), precision in Figure 6(c), and F1-score in Figure 6(d) for both datasets.

Figure 6 reveals that detection accuracy was high for both datasets, with the UNSW-NB15 dataset consistently outperforming the NSL-KDD dataset across all algorithms. A similar trend was observed for the Recall and Precision metrics. For the F1-score, the UNSW-NB15 dataset achieved higher values across all classification algorithms, except for the DL technique MLP, where both datasets reached an equal F1-score of 99%. The comparison with previous studies is summarized in Table 6.

Table 5. Performance parameters for the UNSW-NB15 dataset

| Classification algorithm | Accuracy | | Recall | | Precision | | F1-score | |
|---|---|---|---|---|---|---|---|---|
| | UNSW-NB15 | NSL-KDD | UNSW-NB15 | NSL-KDD | UNSW-NB15 | NSL-KDD | UNSW-NB15 | NSL-KDD |
| NB | 0.958 | 0.901 | 0.96 | 0.9 | 0.96 | 0.9 | 0.96 | 0.9 |
| KNN | 0.974 | 0.953 | 0.98 | 0.95 | 0.98 | 0.95 | 0.98 | 0.95 |
| RF | 0.997 | 0.978 | 0.99 | 0.98 | 0.99 | 0.98 | 0.99 | 0.98 |
| DT | 0.985 | 0.977 | 0.99 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 |
| MLP | 0.983 | 0.989 | 0.98 | 0.99 | 0.99 | 0.98 | 0.99 | 0.99 |
| LSTM | 0.987 | 0.975 | 0.97 | 0.97 | 0.99 | 0.97 | 0.98 | 0.97 |

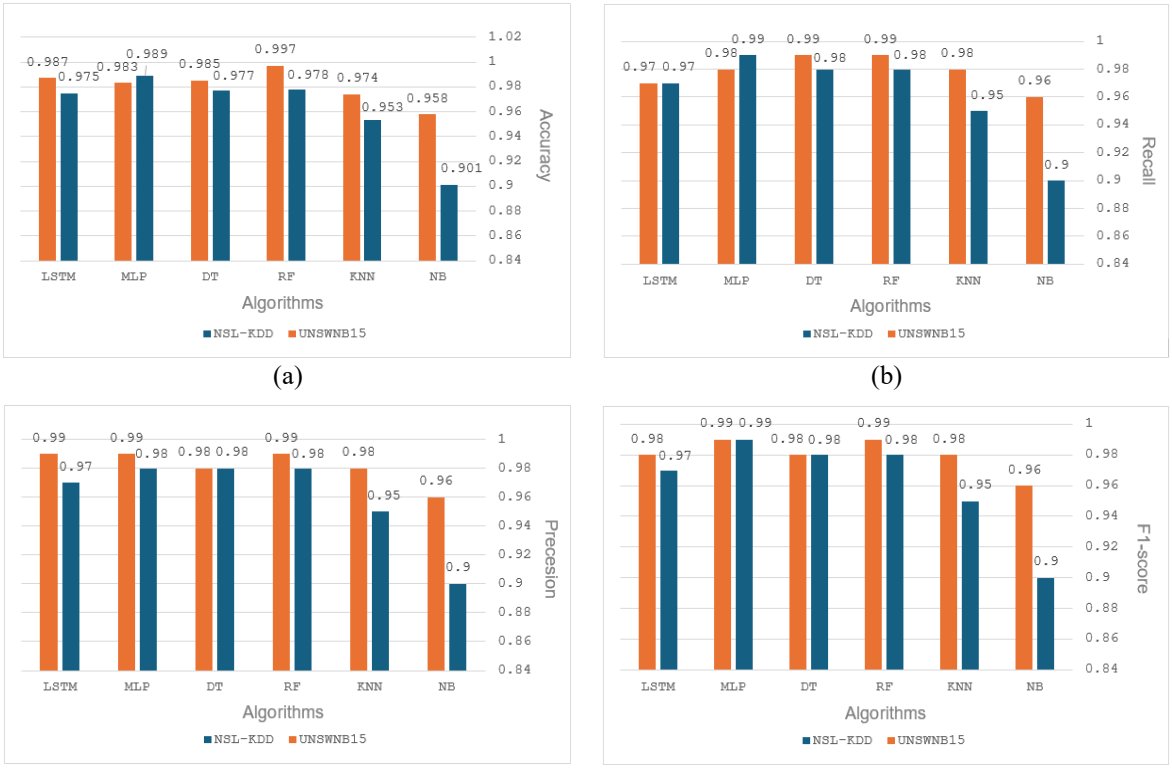


Figure 6. Comparison between algorithms on the x-axis of performance metrics in y-axis: (a) detection accuracy comparison, (b) recall value comparison, (c) precision value comparison, and (d) F1-score value comparison

Table 6 as observed from comparison of the proposed models with previously published works in terms of intrusion detection ratio, we can say that a significant enhancement has been achieved in detection accuracy. For example in [18] they used ANN in NSL-KDD and obtained 95% accuracy while in this study it was 98.9% gotten from NSL-KDD employing MLP. In the same way, decision trees on CICIDS2017 lead to an accuracy of 92.6% [19], Random Forest in our experiments gathered 99.7% on UNSW-NB15. These improvements resulted from the application of sophisticated feature selection methods, and fine-tuned hyperparameters, which ultimately led to enhanced classification and less over fitting.

The findings in Table 6 indicate that the current study achieved superior detection accuracy compared to earlier research efforts. In the first study, neural networks were applied to the NSL-KDD dataset, resulting in a detection accuracy of 95%. The second study utilized DT with the CICIDS2017

dataset, achieving 92.60% accuracy. The third study employed the RF algorithm on the UGR16 dataset, yielding an accuracy of 94%. In contrast, the current study analyzed two datasets, UNSW-NB15 and NSL-KDD, using multiple ML techniques. RF delivered the best accuracy on the UNSW-NB15 dataset, while the DL model with MLP achieved the highest accuracy on the NSL-KDD dataset, surpassing the performance of previous studies.

Table 6. Comparison with previous studies

| Classification algorithm | Dataset | Detection accuracy (%) | Reference |
|---|---|---|---|
| ANN | NSL-KDD | 95% | [19] |
| DT | NSL-KDD | 92.6% | [20] |
| RF | CICID 2017 | 94% | [21] |
| DT, RF, KNN, NB, MLP, LSTM | UGR'16 | 99.7% | This study |
| DT, RF, KNN, NB, MLP, LSTM | UNSW-NB15 | 98.9% | |

There are several limitations to this study even though our results are promising. First, the experiments were performed on two benchmark data sets, so they might not fully reflect the actual network traffic. Second, the deep learning models took a large amount of computation and training time, which came with an inherent restriction for being executed in resource-constrained environments. Also, the applied feature reduction technique could have been more informative and have produced a reduction in the original dataset that lost information. In the future, we plan to mitigate these limitations by including more diverse datasets and investigating lightweight deep learning architectures for real-time applications.

## 5. CONCLUSION

This research employed a diversity of ML and DL methods to identify network intrusions. By evaluating the performance of these methods across two datasets, several key conclusions appeared i) ML algorithms demonstrated strong classification performance ,the MLP, a DL method, achieved the highest accuracy for the NSL-KDD dataset, and the RF algorithm delivered the best results for the UNSW-NB15 dataset, ii) Comparison of techniques while DL techniques like MLP were highly effective, traditional ML algorithms also achieved impressive outcomes, reaffirming their value as viable alternatives in intrusion detection, iii) Broader implications where this research highlights the potential for refining algorithmic approaches by identifying weaknesses and leveraging results to develop new hybrid systems that combine ML and DL techniques.

Recommendations for future research may in dataset improvements to improve datasets by addressing class imbalances and expanding the range of attacks represented, model development to design novel models that integrate diverse ML approaches to improve detection capabilities, real-time adaptation to implement real-time IDS with support from web-based interfaces for dynamic and immediate threat management, and frequent update to conduct regular studies using updated datasets to stay ahead of evolving cyber threats, ensuring that IDS remain effective in the face of new attack methods.

## REFERENCES

[1] R. Thomas and D. Pavithran, "A Survey of intrusion detection models based on NSL-KDD data set," in *2018 Fifth HCT Information Technology Trends (ITT)*, Nov. 2018, pp. 286–291, doi: 10.1109/CTIT.2018.8649498.

[2] J. Zhao, S. Zhang, B. Liu, and B. Cui, "Malware detection using machine learning based on the combination of dynamic and static features," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, Jul. 2018, pp. 1–6, doi: 10.1109/ICCCN.2018.8487459.

[3] M. S. Akhtar and T. Feng, "Malware analysis and detection using machine learning algorithms," *Symmetry*, vol. 14, no. 11, p. 2304, Nov. 2022, doi: 10.3390/sym14112304.

[4] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: a comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, Jan. 2013, doi: 10.1016/j.jnca.2012.09.004.

[5] S. Moualla, K. Khorzom, and A. Jafar, "Improving the performance of machine learning-based network intrusion detection systems on the UNSW-NB15 dataset," *Computational Intelligence and Neuroscience*, vol. 2021, no. 1, Jan. 2021, doi: 10.1155/2021/5557577.

[6] C. Rajathi and P. Rukmani, "Hybrid learning model for intrusion detection system: a combination of parametric and non-parametric classifiers," *Alexandria Engineering Journal*, vol. 112, pp. 384–396, Jan. 2025, doi: 10.1016/j.aej.2024.10.101.

[7] A. Odeh and A. Abu Taleb, "Ensemble-based deep learning models for enhancing IoT intrusion detection," *Applied Sciences*, vol. 13, no. 21, p. 11985, Nov. 2023, doi: 10.3390/app132111985.

[8] A. Aldhaheri, F. Alwahedi, M. A. Ferrag, and A. Battah, "Deep learning for cyber threat detection in IoT networks: a review," *Internet of Things and Cyber-Physical Systems*, vol. 4, pp. 110–128, 2024, doi: 10.1016/j.iotcps.2023.09.003.

[9] 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, p. 79, Mar. 2024, doi: 10.3390/systems12030079.

[10] A. Alharthi, M. Alaryani, and S. Kaddoura, "A comparative study of machine learning and deep learning models in binary and

multiclass classification for intrusion detection systems," *Array*, vol. 26, p. 100406, Jul. 2025, doi: 10.1016/j.array.2025.100406.
[11] V. Shanmugam, R. Razavi-Far, and E. Hallaji, "Addressing class imbalance in intrusion detection: a comprehensive evaluation of machine learning approaches," *Electronics*, vol. 14, no. 1, p. 69, Dec. 2024, doi: 10.3390/electronics14010069.
[12] A. Jain, R. Bagoria, and P. Arora, "An intelligent zero-day attack detection system using unsupervised machine learning for enhancing cyber security," *Knowledge-Based Systems*, vol. 324, p. 113833, Aug. 2025, doi: 10.1016/j.knosys.2025.113833.
[13] S. Muneer, U. Farooq, A. Athar, M. Ahsan Raza, T. M. Ghazal, and S. Sakib, "A critical review of artificial intelligence based approaches in intrusion detection: a comprehensive analysis," *Journal of Engineering*, vol. 2024, no. 1, Jan. 2024, doi: 10.1155/2024/3909173.
[14] R. Anderson, *Security engineering: a guide to building dependable distributed systems*. Indianapolis: John Wiley & Sons, 2010.
[15] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, Mar. 2014, doi: 10.1016/j.eswa.2013.08.066.
[16] M. Ozkan-Okay, R. Samet, O. Aslan, and D. Gupta, "A comprehensive systematic literature review on intrusion detection systems," *IEEE Access*, vol. 9, pp. 157727–157760, 2021, doi: 10.1109/ACCESS.2021.3129336.
[17] T. Sowmya and E. A. Mary Anita, "A comprehensive review of AI based intrusion detection system," *Measurement: Sensors*, vol. 28, p. 100827, Aug. 2023, doi: 10.1016/j.measen.2023.100827.
[18] K. A. Taher, B. Mohammed Yasin Jisan, and M. M. Rahman, "Network intrusion detection using supervised machine learning technique with feature selection," in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, Jan. 2019, pp. 643–646, doi: 10.1109/ICREST.2019.8644161.
[19] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-stage optimized machine learning framework for network intrusion detection," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1803–1816, Jun. 2021, doi: 10.1109/TNSM.2020.3014929.
[20] R. Magán-Carrión, D. Urda, I. Díaz-Cano, and B. Dorronsoro, "Towards a reliable comparison and evaluation of network intrusion detection systems based on machine learning approaches," *Applied Sciences*, vol. 10, no. 5, p. 1775, Mar. 2020, doi: 10.3390/app10051775.
[21] G. A. MM, J. N. K. S, U. M. R, and M. R. TF, "An efficient SVM based DEHO classifier to detect DDoS attack in cloud computing environment," *Computer Networks*, vol. 215, p. 109138, Oct. 2022, doi: 10.1016/j.comnet.2022.109138.
[22] E. Mushtaq, A. Zameer, M. Umer, and A. A. Abbasi, "A two-stage intrusion detection system with auto-encoder and LSTMs," *Applied Soft Computing*, vol. 121, p. 108768, May 2022, doi: 10.1016/j.asoc.2022.108768.
[23] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, Nov. 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.
[24] S. Choudhary and N. Kesswani, "Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT," *Procedia Computer Science*, vol. 167, pp. 1561–1573, 2020, doi: 10.1016/j.procs.2020.03.367.
[25] B. M. Serinelli, A. Collen, and N. A. Nijdam, "Training guidance with KDD cup 1999 and NSL-KDD data sets of ANIDINR: anomaly-based network intrusion detection system," *Procedia Computer Science*, vol. 175, pp. 560–565, 2020, doi: 10.1016/j.procs.2020.07.080.
[26] A. E. Cil, K. Yildiz, and A. Buldu, "Detection of DDoS attacks with feed forward based deep neural network model," *Expert Systems with Applications*, vol. 169, p. 114520, May 2021, doi: 10.1016/j.eswa.2020.114520.
[27] G. C. Amaizu, C. I. Nwakanma, S. Bhardwaj, J. M. Lee, and D. S. Kim, "Composite and efficient DDoS attack detection framework for B5G networks," *Computer Networks*, vol. 188, p. 107871, Apr. 2021, doi: 10.1016/j.comnet.2021.107871.
[28] A. Basati and M. M. Faghih, "PDAE: efficient network intrusion detection in IoT using parallel deep auto-encoders," *Information Sciences*, vol. 598, pp. 57–74, Jun. 2022, doi: 10.1016/j.ins.2022.03.065.
[29] S. Buyrukoglu, "Promising cryptocurrency analysis using deep learning," in *2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Oct. 2021, pp. 372–376, doi: 10.1109/ISMSIT52890.2021.9604721.
[30] P. TS and P. Shrinivasacharya, "Evaluating neural networks using Bi-Directional LSTM for network IDS (intrusion detection systems) in cyber security," *Global Transitions Proceedings*, vol. 2, no. 2, pp. 448–454, Nov. 2021, doi: 10.1016/j.gltp.2021.08.017.
[31] M. Al-Shabi, "Design of a network intrusion detection system using complex deep neuronal networks," *International Journal of Communication Networks and Information Security*, vol. 13, no. 3, pp. 409–415, 2021.
[32] R. Kruse, S. Mostaghim, C. Borgelt, C. Braune, and M. Steinbrecher, "Multi-layer perceptrons," in *Computational intelligence: a methodological introduction*, Springer, 2022, pp. 53–124.
[33] T. D. U. T. CONVOLUTIONAL, "Real-time anomaly detection in Internet of Things devices using temporal convolutional network," *Journal of Theoretical and Applied Information Technology*, vol. 101, no. 16, 2023.
[34] D. Božić, B. Runje, D. Lisjak, and D. Kolar, "Metrics related to confusion matrix as tools for conformity assessment decisions," *Applied Sciences*, vol. 13, no. 14, p. 8187, Jul. 2023, doi: 10.3390/app13148187.

## BIOGRAPHIES OF AUTHORS

**Mohammed Al-Shabi** is an associate professor in the Department of MIS, Faculty of Business Administration, at Taibah University, KSA. He received his bachelor's degree from the computer science department at University at Iraq (1997). He received his master's degree in computer science from Putra Malaysia University at 2002, and Ph.D. (computer science) from Putra Malaysia University, Malaysia (2006). His research interests are mainly directed to network security, data protection, and cloud security. In addition to data protection and privacy, developing new protocols, and using artificial intelligence in cybersecurity, wireless security, cryptography, UML, stenography multistage interconnection network, parallel computing and apply mathematic. He can be contacted at email: mshaby@taibahu.edu.sa.

**Anmar Abuhamdah** received the B.Sc. degree in computer science from Princess Sumaya University for Technology, Jordan, in 2003, and his M.Sc. degree in intelligent systems from Utara University, Malaysia, in 2007, and the Ph.D. degree in computer science from the National University of Malaysia, Malaysia, in 2011. He is currently an associate professor at Taibah University, Kingdom of Saudi Arabia, since 2018. His research interests are mainly directed to metaheuristics and combinatorial optimization problems including timetabling, routing, quadratic, and rostering. His research interests include the applications of artificial intelligence, including deep learning, evolutionary and heuristic optimization techniques, operation and control, text classification, feature selection prediction models. He can be contacted at email: aabuhamdah@taibahu.edu.sa.

**Malek Alzaqebah** received the B.Sc. degree in computer science from Al Balqa' Applied University, Jordan, in 2006, and his M.Sc. degree in information technology from Utara University, Malaysia, in 2008, and the Ph.D. degree in computer science from the National University of Malaysia, Malaysia in 2012. He is currently an assistant professor in the department mathematics, college of science at Imam Abdulrahman Bin Faisal University. His research interests are mainly directed to metaheuristics and combinatorial optimization problems including course and exam timetabling, vehicle routing, travelling salesman, knapsack, and nurse rostering problems. He can be contacted at email: maafehaid@iau.edu.sa.