Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

TOPIC: Deep Learning Approach for Network Intrusion Detection and Multi-Class Attack Classification Using UNSW-NB15 Dataset

Akpasam Ekanem

akpasamekanem@aksu.edu.ng

Akwa Ibom State University

Philip Asuquo

University of Uyo

Simeon Ozoumba

University of Uyo

Bliss Stephen

University of Uyo

Enyinihi Johnson

Akwa Ibom State University

Article

Keywords: Intrusion Detection System (IDS), Deep Learning, RNN-LSTM, CNN, ANN, UNSW-NB15, Cybersecurity, Multiclass Classification

Posted Date: July 8th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8387299/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

TOPIC: Deep Learning Approach for Network Intrusion Detection and Multi-Class Attack Classification Using UNSW-NB15 Dataset

Akpasam Ekanem, Philip Asuquo, Simeon Ozoumba, Bliss Stephen, Enyinihi H. Johnson

Abstract
The increasing sophistication of cyber threats has created a growing need for intelligent intrusion detection systems capable of accurately identifying malicious network activities. Traditional intrusion detection approaches often struggle to detect complex and previously unseen attack patterns, thereby motivating the adoption of deep learning techniques for network security applications. This study presents a comparative deep learning framework for network intrusion detection and multiclass attack classification using the UNSW-NB15 dataset. The proposed framework incorporates feature selection, dataset partitioning, feature scaling, and class balancing using the Synthetic Minority Oversampling Technique (SMOTE) to improve model training and classification performance. Three deep learning architectures, namely Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), and Recurrent Neural Networks with Long Short-Term Memory (RNN-LSTM), were developed and evaluated using multiple performance metrics, including accuracy, precision, recall, F1-score, confusion matrix analysis, and Receiver Operating Characteristic Area Under the Curve (ROC-AUC). Experimental results demonstrated that all three models achieved strong intrusion detection performance. Among the evaluated architectures, the RNN-LSTM model achieved the best overall performance, recording an accuracy of 97.42%, precision of 97.45%, recall of 97.42%, and F1-score of 97.42%. Comparative analysis further revealed that the RNN-LSTM architecture outperformed the ANN and CNN models in multiclass intrusion classification on the UNSW-NB15 dataset. The findings demonstrate the effectiveness of deep learning techniques for network intrusion detection and highlight the potential of RNN-LSTM architectures for improving cybersecurity monitoring and attack classification in modern network environments.

Keywords: Intrusion Detection System (IDS), Deep Learning, RNN-LSTM, CNN, ANN, UNSW-NB15, Cybersecurity, Multiclass Classification

1.1 Introduction

The increasing dependence on internet-connected systems for communication, finance, healthcare, e-commerce, cloud computing, and critical infrastructure has significantly increased the frequency and sophistication of cyber threats [1], [2]. Modern cyberattacks often involve multiple malicious activities such as reconnaissance, exploitation, privilege escalation, malware propagation, and denial-of-service attacks, which can severely compromise network security, privacy, and service availability [3], [4]. Consequently, the development of intelligent intrusion detection systems (IDSs) capable of accurately identifying malicious network behavior has become an important research challenge [5].

Traditional intrusion detection techniques, including signature-based and statistical approaches, often struggle to detect evolving and previously unseen attack patterns due to their limited adaptability and dependence on predefined rules [6], [7]. As a result, machine learning and deep learning approaches have gained significant attention because of their ability to automatically learn complex nonlinear patterns from high-dimensional network traffic data [8], [9]. Deep learning models such as Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and

Recurrent Neural Networks (RNNs) have demonstrated promising performance in intrusion detection and malicious traffic classification tasks [10]–[13].

Although several studies have applied deep learning techniques to intrusion detection using benchmark datasets such as UNSW-NB15, there remains a need for systematic comparative evaluation of different deep learning architectures under consistent preprocessing, balancing, and evaluation conditions [14], [15]. Furthermore, intrusion detection datasets commonly exhibit class imbalance, feature heterogeneity, and varying attack distributions, which continue to affect classification performance across minority attack categories [16], [17].

Recent intrusion detection research has increasingly explored advanced AI-driven architectures for cybersecurity applications. Dai et al. [18] investigated machine learning techniques for zero-day intrusion detection using unseen traffic patterns, while Sana et al. [19] explored Vision Transformer-based intrusion detection frameworks for IoT cybersecurity environments. Similarly, Yang et al. [20] proposed a hybrid large language model architecture for web attack detection. Despite these advances, variations in preprocessing pipelines, validation strategies, and evaluation protocols continue to make direct comparison between intrusion detection models difficult.

This study therefore presents a comparative deep learning framework for network intrusion detection and multiclass attack classification using the UNSW-NB15 dataset. The study evaluates the performance of ANN, CNN, and RNN-LSTM models using standardized preprocessing, feature scaling, controlled SMOTE balancing, and multi-metric evaluation procedures. The contributions of this work include:

- A controlled comparative evaluation of ANN, CNN, and RNN-LSTM architectures for multiclass intrusion classification;
- An improved preprocessing pipeline involving feature cleaning, feature scaling, stratified train-test splitting, and controlled SMOTE balancing applied exclusively to the training dataset;
- Comprehensive evaluation using accuracy, precision, recall, F1-score, ROC-AUC and confusion matrices;
- Analysis of computational performance, classification behavior, and practical trade-offs across different attack categories and deep learning architectures.

2.0 Related work

Network intrusion detection has attracted significant research attention due to the increasing sophistication of cyber threats and the growing need for intelligent cybersecurity solutions. Existing studies have explored a wide range of machine learning, deep learning, and hybrid approaches for detecting and classifying malicious network activities. Benchmark datasets such as KDD99, NSL-KDD, CICIDS, and UNSW-NB15 have been widely employed to evaluate the effectiveness of these techniques. This section reviews relevant literature on machine learning-based intrusion detection, deep learning architectures, class imbalance handling techniques, and recent advances in transformer-based cybersecurity frameworks. The review also highlights existing research gaps that motivate the present study.

2.1 Intrusion Detection System

Intrusion Detection Systems (IDSs) play a critical role in protecting computer networks and information systems against unauthorized access, malicious activities, and cyberattacks. Traditional IDS techniques are generally categorized into signature-based and anomaly-based approaches. Signature-based systems detect known attack patterns using predefined rules and attack signatures, while anomaly-based systems identify deviations from normal network behavior to detect potentially unknown threats [5], [6].

Although signature-based IDSs are effective in detecting previously known attacks with relatively low false alarm rates, they often struggle to identify zero-day attacks and evolving cyber threats due to their dependence on continuously updated signature databases [1]. Similarly, statistical and rule-based anomaly detection techniques frequently encounter challenges when processing large-scale and highly dynamic network traffic environments [12].

2.2 Machine Learning-Based Intrusion Detection

To overcome the limitations of traditional intrusion detection techniques, machine learning (ML) approaches have been widely adopted for automated cyberattack detection and classification. Machine learning models such as Decision Trees, Support Vector Machines (SVM), Random Forests, Naïve Bayes, and Extreme Gradient Boosting (XGBoost) have demonstrated promising performance in intrusion detection applications due to their ability to learn complex decision boundaries from network traffic data [15], [21].

Dai et al. [18] proposed a machine learning-based intrusion detection framework capable of identifying zero-day attacks from previously unseen network traffic patterns. Their study demonstrated the potential of adaptive machine learning models for detecting emerging cyber threats. Similarly, several studies have reported strong classification performance using ensemble learning approaches; however, many of these methods remain sensitive to feature engineering, class imbalance, and dataset-specific characteristics [14].

Despite their effectiveness, traditional machine learning approaches often require extensive manual feature selection and may experience performance degradation when handling highly complex and high-dimensional network traffic data [11].

2.3 Deep Learning-Based Intrusion Detection

Deep learning techniques have recently gained significant attention in cybersecurity because of their ability to automatically learn hierarchical feature representations from raw or minimally processed data. Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) have been successfully applied to intrusion detection tasks using benchmark datasets such as NSL-KDD, CICIDS2017, and UNSW-NB15 [10],[22],[23].

Javaid et al. [13] demonstrated the effectiveness of deep neural networks for network intrusion detection by automatically learning discriminative traffic features without extensive manual feature engineering. CNN-based approaches have also shown strong performance in intrusion classification by extracting local feature patterns through convolutional operations [14]. Similarly, RNN and Long Short-Term Memory (LSTM) architectures have been employed to capture complex feature dependencies within network traffic representations [24].

Ferrag et al. [14] conducted a comprehensive review of deep learning approaches for cybersecurity applications and reported that deep learning architectures generally outperform many traditional machine learning methods when sufficient training data are available. However, deep learning models often require substantial computational resources and may suffer from overfitting, particularly when trained on imbalanced intrusion datasets [14].

2.4 Recent AI-Driven and Transformer-Based Intrusion Detection

Recent advances in artificial intelligence have led to the emergence of transformer-based and hybrid AI architectures for intrusion detection. Sana et al. [19] proposed a Vision Transformer-based intrusion anomaly detection framework for Internet of Things (IoT) environments, demonstrating the capability of transformer architectures to learn complex network traffic representations and long-range feature interactions.Similarly, Yang et al. [20] introduced a hybrid intrusion detection framework combining large language models, autoencoders, and multilayer perceptrons for web attack detection. Their results highlighted the growing potential of foundation models and generative AI technologies in cybersecurity applications. Furthermore, Por et al. [25] conducted a systematic literature review on AI-based methods for zero-day attack detection and highlighted major challenges related to data availability, algorithmic complexity, generalization capability, and real-time deployment of intelligent intrusion detection systems.

Although these advanced architectures have achieved promising results, differences in preprocessing procedures, dataset selection, balancing strategies, and evaluation protocols continue to make direct comparison among intrusion detection models difficult [14].

2.5 Recent Advances in Deep Learning and Transformer-Based Intrusion Detection

Recent intrusion detection research has increasingly focused on addressing class imbalance, feature heterogeneity, zero-day attack detection, and the limitations of conventional deep learning architectures. Several studies have investigated the use of oversampling strategies, hybrid neural architectures, and transformer-based models to improve intrusion detection performance on highly imbalanced benchmark datasets such as UNSW-NB15 and CICIDS2017 [16][17].

Ullah et al. [26] proposed IDS-INT, a transformer-based transfer learning framework designed specifically for imbalanced network traffic classification. Their study demonstrated that transformer architectures can effectively learn complex feature interactions while improving minority attack detection performance. Similarly, Xi et al. [27] introduced a multi-scale transformer-based intrusion detection model capable of capturing both local and global traffic representations through multi-scale feature extraction mechanisms.

Recent Vision Transformer (ViT) approaches have also shown promising performance in intrusion anomaly detection. Sana et al. [19] proposed a Vision Transformer-based framework for IoT cybersecurity environments and demonstrated improved anomaly detection capability through enhanced representation learning. Likewise, Zhou et al. [28] developed HiViT-IDS, an efficient Vision Transformer intrusion detection system that achieved strong classification performance while reducing computational overhead.

Class imbalance remains one of the most significant challenges in network intrusion detection. Chen et al. [29] investigated imbalance-aware preprocessing strategies for the UNSW-NB15 dataset and reported that controlled balancing techniques substantially improved minority attack classification performance. Similarly, Shanmugam et al. [30] conducted a comparative study on

class imbalance handling methods and emphasized the importance of resampling techniques such as SMOTE for improving F1-score and minority class detection capability.

Several recent studies have combined deep learning architectures with balancing mechanisms to improve intrusion detection robustness. Musthafa et al. [31] proposed an intrusion detection framework that integrates balanced class distribution techniques, feature selection, and ensemble machine learning methods for IoT security applications. Their study demonstrated that combining SMOTE-based balancing and feature selection substantially improved intrusion classification performance on benchmark datasets including UNSW-NB15, while Choudhary and Kesswani [32] analyzed the NSL-KDD, KDD-Cup'99, and UNSW-NB15 datasets using deep learning approaches and highlighted the importance of dataset characteristics, preprocessing strategies, and feature representation in intrusion detection performance.

More recently, transformer-enhanced and hybrid AI-based intrusion detection systems have emerged as promising research directions. Yang et al. [20] introduced a hybrid framework combining large language models, autoencoders, and multilayer perceptrons for web attack detection. Furthermore, recent studies have investigated transformer-based explainable intrusion detection systems [33], self-supervised transformer learning frameworks [34], and federated transformer-based intrusion detection architectures [35] to improve adaptability, interpretability, and deployment scalability in modern cybersecurity environments.

Despite these advances, differences in dataset preprocessing procedures, train-test partitioning strategies, balancing methods, evaluation metrics, and validation protocols continue to make direct comparison among intrusion detection models difficult. Consequently, there remains a need for controlled comparative evaluation of representative deep learning architectures under unified preprocessing and evaluation conditions.

2.6 Research Gap

Recent advances in intrusion detection research have demonstrated the effectiveness of machine learning, deep learning, transformer-based architectures, explainable artificial intelligence, and federated learning frameworks for identifying cyber threats in modern network environments [14], [26]–[35]. Studies such as Dai et al. [18], Sana et al. [19], Xi et al. [27], AL-Nomasy et al. [33], and Abd Elaziz et al. [35] have shown that advanced learning architectures can significantly improve intrusion detection performance through enhanced feature representation, transformer-based learning mechanisms, explainable decision-making, and privacy-preserving distributed training.

Despite these advances, several challenges remain. First, many existing studies focus primarily on introducing novel architectures while providing limited comparative evaluation against representative deep learning models under identical experimental conditions. Second, differences in preprocessing pipelines, feature engineering strategies, balancing procedures, train-test partitioning methods, and evaluation metrics often make direct comparison among proposed intrusion detection models difficult [14]. Third, class imbalance remains a persistent challenge in benchmark intrusion detection datasets, frequently leading to degraded detection performance for minority attack categories despite high overall classification accuracy [16], [17].

Furthermore, while recent transformer-based and hybrid AI approaches have reported promising performance, many studies emphasize architectural innovation rather than systematically analyzing the relative classification behavior, computational characteristics, and performance

trade-offs of widely adopted deep learning architectures such as ANN, CNN, and RNN-LSTM models under a unified evaluation framework.

Therefore, there remains a need for a controlled comparative evaluation of representative deep learning architectures using standardized preprocessing, controlled class balancing, and consistent validation procedures. This study addresses this gap by conducting a systematic comparative analysis of ANN, CNN, and RNN-LSTM architectures for multiclass intrusion detection using the UNSW-NB15 dataset. The proposed framework incorporates standardized preprocessing, stratified train-test partitioning, SMOTE balancing applied exclusively to the training dataset and comprehensive evaluation using accuracy, precision, recall, F1-score, ROC-AUC, confusion matrices, and computational performance analysis.

3.0 PROPOSED MODELS

This section discusses the deep learning approaches and the proposed models implementation. The hardware materials used in this research includes Apple Laptop with macOS Sonoma 14, 16 gigabytes of random-access memory (GB), internal memory of 512GB with Apple M3 Chip processor and HP 2000 notebook laptop, core i3 processor, 8GB of RAM, Ver.22H2, hard disk memory size of 500 GB with intel high-definition Graphics 4000 adapter. The software material technologies and tools includes Jupyter Notebook and Laboratory, Tensorflow, Keras, Matplotlib, Scikit-learn library and Amazon web service which was used for training the dataset.

3.1 Artificial Neural Network (ANN) Models

Following data preprocessing, feature encoding, and normalization, the dataset was partitioned into training and testing subsets using a stratified train-test split. To prevent information leakage, Synthetic Minority Oversampling Technique (SMOTE) balancing was applied exclusively to the training subset, while the testing dataset remained completely untouched throughout model development and evaluation. The balanced training data were subsequently used to train the Artificial Neural Network (ANN) classifier.

For multiclass intrusion classification, the target labels were transformed into categorical representations using one-hot encoding through the to_categorical function available in the TensorFlow/Keras framework. The ANN model was implemented using a feedforward sequential architecture consisting of fully connected dense layers. The architecture as presented in Figure 1 comprised an input layer with 12 neurons using the Rectified Linear Unit (ReLU) activation function, followed by a hidden layer containing 8 ReLU-activated neurons. The output layer consisted of 10 neurons corresponding to the intrusion classes present in the UNSW-NB15 dataset and employed the SoftMax activation function to generate class probability distributions.

The ANN architecture was selected because of its relatively low computational complexity and its ability to learn nonlinear relationships among network traffic features. As one of the most widely adopted baseline deep learning models, ANN provides a useful benchmark for evaluating the relative performance of more complex architectures such as CNN and RNN-LSTM under identical experimental conditions.

The model was trained using the Adam optimization algorithm with a learning rate of 0.001 and optimized using the categorical cross-entropy loss function, which is appropriate for multiclass classification problems. Training was conducted for 100 epochs with a batch size of 32. Model

performance was evaluated using multiple classification metrics, including accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrix analysis.

The training history, including loss and performance metrics recorded across epochs, was retained for convergence analysis and comparative evaluation with the CNN and RNN-LSTM architectures.

For the ANN model, each neuron computes a weighted linear combination followed by a nonlinear activation function [36]:

$$Z = \sum_{i=1}^{n} \mathrm{WiXi} + b$$

where:

- Xi denotes the i^{th} input feature,
- Wi represents the corresponding connection weight,
- b is the bias term,
- and z is the weighted input to the neuron.

The Rectified Linear Unit (ReLU) activation function is defined as:

f(x) = max(0,x)

The output layer uses the SoftMax activation function for multiclass classification:

$$\mathrm{P(y=j|x)}=\frac{e^{zj}}{\sum_{k=1}^{K} e^{zk}}$$

where:

- P(y=j|x) is the probability that input sample x belongs to class j,
- zj is the output score associated with class j,
- and K denotes the total number of target classes.

The SoftMax function converts the output scores into a probability distribution whose values sum to one, making it suitable for multiclass intrusion classification.

For categorical cross-entropy loss

$$\mathrm{L}= -\sum_{i=1}^{K} yi\log(\hat{y}i)$$

where:

- yi is the true class label,
- $\hat{y}i$ is the predicted probability,

- and K is the number of classes.

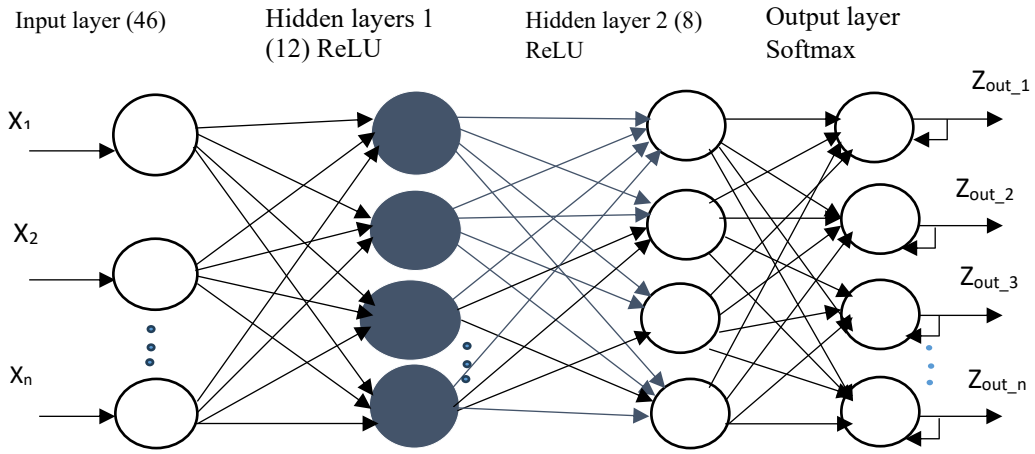


Figure 1: Architecture of the proposed ANN model for multiclass intrusion classification.

3.2 Convolutional Neural Network (CNN) Model

Following data preprocessing, feature encoding, normalization, and controlled class balancing using SMOTE on the training subset, a Convolutional Neural Network (CNN) architecture was developed to perform multiclass intrusion classification on the UNSW-NB15 dataset. CNNs are widely used for automatic feature extraction because of their ability to learn hierarchical representations and identify localized feature patterns within structured input data.

To facilitate one-dimensional convolution operations, the normalized feature vectors were reshaped into a one-dimensional format compatible with the Conv1D layer. The CNN architecture, illustrated in Figure 2, begins with a one-dimensional convolutional layer consisting of 64 filters and a kernel size of 3. The Rectified Linear Unit (ReLU) activation function was employed to introduce nonlinearity and enable the extraction of discriminative feature representations from the input traffic data.

A max-pooling layer with a pool size of 2 was subsequently applied to reduce the dimensionality of the extracted feature maps while preserving the most informative features. This operation also contributes to computational efficiency and helps reduce the risk of overfitting. The resulting feature maps were then flattened and passed to a fully connected dense layer containing 64 neurons activated using the ReLU function.

The output layer consisted of 10 neurons corresponding to the intrusion classes present in the UNSW-NB15 dataset and employed the SoftMax activation function to generate probability distributions for multiclass classification. The CNN architecture was selected because of its capability to automatically learn localized feature relationships within network traffic representations while maintaining a moderate computational complexity relative to recurrent architectures.

The model was trained using the Adam optimization algorithm with a learning rate of 0.001 and optimized using the categorical cross-entropy loss function. Training was conducted for 100 epochs with a batch size of 32. Performance evaluation was performed using multiple classification metrics, including accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrix analysis. The CNN configuration is presented in Table 1.

The training history, including loss and classification performance across epochs, was retained for convergence analysis and comparative evaluation against the ANN and RNN-LSTM architectures. This enabled a consistent assessment of predictive performance, computational characteristics, and classification behavior across the evaluated deep learning models.

The CNN model employs one-dimensional convolution operations to automatically extract localized feature representations from the transformed network traffic vectors. The convolution operation is expressed as [37]:

$$s(t) = (x * w)(t) = \sum_{\propto} X(a)w(t - a)$$

where x represents the input feature vector, w denotes the convolution kernel, and s(t) represents the resulting feature map. Following convolution, a max-pooling operation was applied to reduce feature dimensionality and computational complexity while preserving the most informative features. The hyperparameter settings adopted in this study were selected based on empirical performance stability, computational efficiency considerations, and configurations commonly reported in related intrusion detection literature.

Table 1; CNN Architecture Configuration

Layer	**Configuration**	**Output Shape**
Input Layer	46 Features	(46,1)
Conv1D	64 Filters, Kernel Size = 3, ReLU	(44,64)
MaxPooling1D	Pool Size = 2	(22,64)
Flatten	—	(1408)
Dense	64 Neurons, ReLU	(64)
Output Layer	10 Neurons, SoftMax	(10)

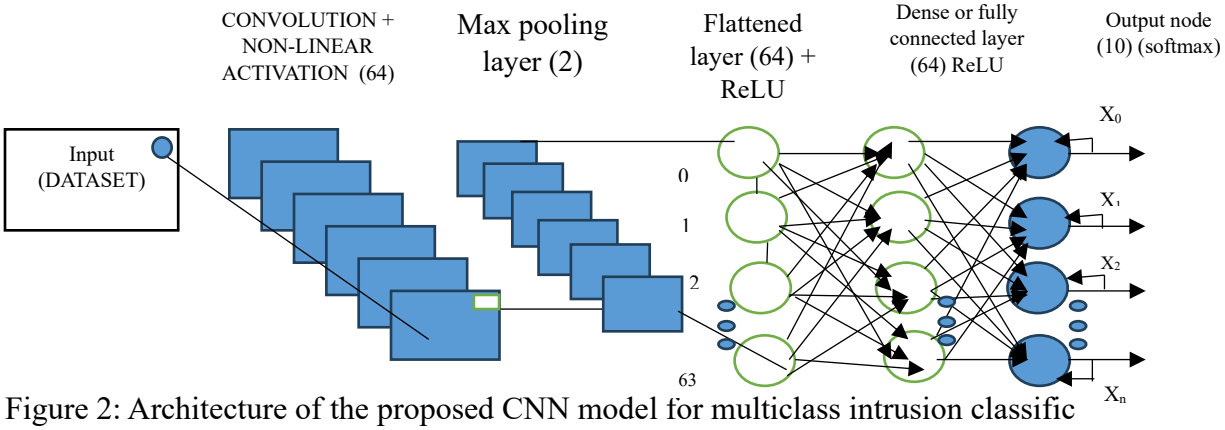


Figure 2: Architecture of the proposed CNN model for multiclass intrusion classific

3.3 Recurrent Neural network with Long Short-Term Memory (RNN-LSTM) Architecture for UNSW-NB15

Following data preprocessing, feature encoding, normalization, and controlled class balancing using SMOTE on the training subset, a Recurrent Neural Network with Long Short-Term Memory (RNN-LSTM) architecture was implemented for multiclass intrusion classification using the UNSW-NB15 dataset. LSTM networks are widely recognized for their ability to learn complex sequential and contextual relationships through gated memory mechanisms and have been successfully applied in various intrusion detection applications.

To enable compatibility with the LSTM layer, the normalized feature vectors were reshaped into a three-dimensional tensor representation of shape (N, F, 1), where N represents the number of samples and F denotes the number of input features. In this configuration, the ordered feature dimensions were treated as a structured sequence representation to facilitate deep feature learning within the LSTM architecture. It is important to note that the reshaped representation does not imply explicit temporal attack progression; rather, it provides a consistent input format for evaluating the feature-learning capability of the LSTM model on the UNSW-NB15 dataset.

The architecture, illustrated in Figure 3, begins with a single LSTM layer containing 64 hidden units. The gated memory structure of the LSTM layer enables the model to learn complex dependencies and interactions among the transformed feature representations. The output from the LSTM layer is subsequently passed to a fully connected dense layer consisting of 64 neurons activated using the Rectified Linear Unit (ReLU) function. This dense layer introduces additional nonlinear representation learning capability and enhances the model's ability to capture complex feature relationships.

The final classification layer consists of 10 neurons corresponding to the intrusion classes present in the UNSW-NB15 dataset and employs the SoftMax activation function to generate class probability distributions for multiclass classification. The RNN-LSTM architecture was selected because of its ability to model complex feature interactions through memory-based learning mechanisms and to provide a comparative assessment against ANN and CNN architectures under identical preprocessing and evaluation conditions.

The model was trained using the Adam optimization algorithm with a learning rate of 0.001 and optimized using the categorical cross-entropy loss function. Training was conducted for 100 epochs using a batch size of 64. Model performance was evaluated using multiple metrics, including accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrix analysis. The architecture configuration of RNN-LSTM is presented in table 2.

The training history, including loss and classification performance recorded across epochs, was retained for convergence analysis and comparative evaluation against the ANN and CNN architectures. This enabled a consistent assessment of classification effectiveness, computational characteristics, and performance trade-offs across all evaluated deep learning models [38].

The equations used are presented below:
RNN hidden state:

$$h_t = f(W_h h_{t-1} + W_x \mathcal{X}_t + b)$$

and then present the LSTM update equations as a compact group:
Forget gate:

$$f_t=\sigma\ (W_f[h_{t-1},\ \varkappa_t] + b_f)$$

Input gate:

$$\acute{\imath}_t=\sigma\ (W_i[h_{t-1},\ \varkappa_t] + bi)$$

Cell state:

$$C_t = f_t \odot C_{t-1}+\acute{\imath}_t \odot \acute{C}_t$$

Output gate:

$$O_t = \sigma\ (W_o[h_{t-1},x_t] + b_o)$$

Final hidden state:

$$h_t = o_t \odot \tanh\ (C_t)$$

These gating mechanisms enable the LSTM architecture to selectively retain, update, and propagate information through the network, thereby supporting the learning of complex feature dependencies within the transformed intrusion data representation.

The sequential representation employed in this study does not correspond to temporal attack progression or chronological network events. Rather, the feature vector was reshaped to facilitate the application of the LSTM architecture for learning inter-feature dependencies within the tabular intrusion detection dataset.

Table 2: RNN-LSTM Architecture Configuration

Layer	**Configuration**	**Output Shape**
Input Layer	Reshaped Feature Vector	(46,1)
LSTM Layer	64 Units	(64)
Dense Layer	64 Neurons, ReLU	(64)
Output Layer	10 Neurons, SoftMax	(10)

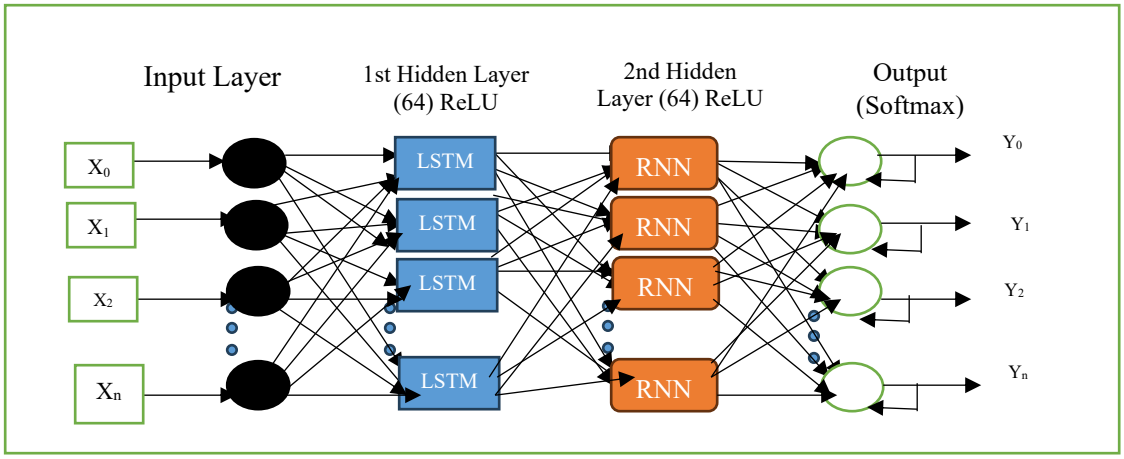


Figure 3: Architecture of the proposed RNN-LSTM model for multiclass intrusion classification

3.4 Hyperparameter Sensitivity Analysis

Preliminary experiments were conducted to evaluate the effect of selected hyperparameters on model stability and performance. For the RNN-LSTM model, configurations with 32, 64, and 128 hidden units were explored. Although larger hidden dimensions slightly improved training accuracy, the 64-unit configuration provided a more stable balance between predictive performance and computational efficiency.

Similarly, CNN kernel sizes of 3 and 5 were evaluated. The kernel size of 3 demonstrated slightly better convergence stability and lower computational overhead under the adopted dataset representation.

Early stopping and dropout regularization were also explored during preliminary experiments; however, fixed 100-epoch training without aggressive regularization produced more stable comparative behavior across the evaluated architectures under the adopted preprocessing framework.

4.0 **UNSW-NB15 Dataset**

The UNSW-NB15 dataset was selected for this study because it provides a modern and comprehensive benchmark for evaluating network intrusion detection systems (NIDSs). Unlike earlier benchmark datasets such as KDD99 and NSL-KDD, which have been criticized for issues including redundant records, outdated traffic patterns, and limited representation of contemporary cyber threats, UNSW-NB15 was specifically developed to reflect modern network environments and attack behaviors [34]–[36].

The dataset was generated at the Cyber Range Laboratory of the Australian Centre for Cyber Security (ACCS) using the IXIA PerfectStorm traffic generation platform [38], [39]. A combination of realistic normal network activities and synthetic attack scenarios was created to simulate contemporary network traffic conditions. Approximately 100 GB of raw network traffic was captured using the tcpdump tool [40], while feature extraction was performed using Argus and Bro-IDS (now known as Zeek) [41], [42].

The UNSW-NB15 dataset contains 49 network traffic features and one target class attribute. These features describe various characteristics of network flows, including packet-level, protocol-level, content-based, and statistical traffic properties. The dataset includes both normal traffic and malicious activities, making it suitable for supervised intrusion detection and multiclass attack classification tasks.

A major advantage of the UNSW-NB15 dataset is its inclusion of nine contemporary attack categories alongside normal network traffic. The attack classes comprise Fuzzers, Analysis, Backdoors, Denial-of-Service (DoS), Exploits, Generic attacks, Reconnaissance, Shellcode, and Worms. This diversity enables comprehensive evaluation of intrusion detection models across multiple attack types and varying traffic behaviors.

Furthermore, the dataset exhibits class imbalance characteristics that closely resemble real-world cybersecurity environments, where certain attack categories occur far less frequently than others. This property makes UNSW-NB15 particularly suitable for investigating the effectiveness of data balancing techniques such as the Synthetic Minority Oversampling Technique (SMOTE) and evaluating the robustness of deep learning architectures under realistic classification conditions.

Given its modern attack representation, diverse feature set, multiclass attack structure, and widespread adoption in intrusion detection research, the UNSW-NB15 dataset provides an appropriate benchmark for the comparative evaluation of ANN, CNN, and RNN-LSTM architectures conducted in this study. Table 3 shows the summary of the UNSW-NB15 Dataset

Table 3. Summary of the UNSW-NB15 Dataset

Characteristic	Description
Dataset Name	UNSW-NB15
Number of Features	49
Target Variable	Attack Category
Normal Class	Yes
Attack Categories	9
Total Classes	10 (including normal traffic)
Generated By	ACCS Cyber Range Lab
Traffic Capture Tool	tcpdump
Feature Extraction Tools	Argus, Bro-IDS (Zeek)
Primary Application	Network Intrusion Detection and Multiclass Attack Classification

5.0 Data Preparation and processing

The dataset contained numerous columns where hyphens (”-”) were used as placeholders for missing values. To accurately assess the extent of missing data, these hyphens were replaced with NaN values. This allowed for a clearer analysis of missing value distribution across columns, which was then visualized in a pie chart. The chart highlighted the proportion of missing values contributed by each column relative to the entire dataset. Columns found to be almost entirely empty, contributing close to 100% missing values were subsequently dropped, as they did not provide sufficient information to support effective model development. Let the intrusion dataset be represented as:

$$D = \{(x_i, y_i)\}_{i=1}^{N}$$

And the input space is represented as:

$$X_i \in \mathbb{R}^m$$

where ($X_i \in \mathbb{R}^m$) denotes the feature vector of the (i^{th}) network traffic sample, (y_i) represents the corresponding attack class label, (m) is the number of input features, and (N) is the total number of samples.

The objective of the classification model is to learn a mapping function

$$f: \mathbb{R}^m \rightarrow C$$

where

$$C=\{c_1, c_2, \ldots c_K\}$$

represents the set of intrusion classes, including normal traffic and attack categories, and (K) denotes the total number of target classes.

Prior to model training, the dataset was subjected to a preprocessing pipeline involving data cleaning, feature encoding, feature scaling, train-test partitioning, and class balancing using the Synthetic Minority Oversampling Technique (SMOTE). To prevent data leakage, SMOTE was applied exclusively to the training subset after dataset partitioning.

The processed feature vectors were subsequently used as inputs to the ANN, CNN, and RNN-LSTM architectures for multiclass intrusion classification.

5.1 Feature Selection

Feature selection was performed to reduce data dimensionality, improve computational efficiency, and minimize the risk of model overfitting while retaining the most relevant network traffic characteristics for intrusion detection. Following data cleaning and preprocessing, a subset of features describing connection behavior, traffic volume, packet statistics, protocol information, and DNS-related activities was selected for model development.

The retained predictor variables included src_port and dst_port (source and destination port numbers), proto (communication protocol), duration (connection duration), src_bytes and dst_bytes (bytes transferred from source to destination and vice versa), conn_state (connection state), missed_bytes (number of missed bytes), src_pkts and dst_pkts (packet counts), src_ip_bytes and dst_ip_bytes (IP byte counts), dns_qclass (DNS query class), dns_qtype (DNS query type), and dns_rcode (DNS response code). Categorical attributes such as proto and conn_state were transformed into numerical representations to facilitate model training.

These selected features provide complementary information regarding communication endpoints, protocol usage, traffic intensity, packet exchange behavior, and DNS transaction characteristics. Collectively, they capture important patterns associated with normal and malicious network activities, making them suitable for multiclass intrusion classification.

The type attribute was retained separately as the target variable representing the intrusion class label. The selected feature set was subsequently standardized and used as input to the ANN, CNN, and RNN-LSTM models.

5.2 Data Partitioning

To facilitate model training and unbiased performance evaluation, the dataset was partitioned into training and testing subsets. A stratified train-test split was employed, with 75% of the data allocated for training and 25% reserved for testing.

Stratification was applied to preserve the original class distribution across both subsets, ensuring that each intrusion category was proportionally represented in the training and testing datasets. This approach helps prevent sampling bias and improves the reliability of performance evaluation, particularly in multiclass intrusion classification problems.

To avoid information leakage, dataset partitioning was performed prior to any class balancing operation. Consequently, the Synthetic Minority Oversampling Technique (SMOTE) was applied

exclusively to the training dataset, while the testing dataset remained completely unchanged and was used solely for model evaluation.

5.3 Feature Scaling

The dataset was first partitioned into training and testing subsets. Feature scaling was subsequently performed using the standardization technique, where the scaling parameters were computed exclusively from the training dataset and then applied to both the training and testing datasets. This approach prevents information leakage and ensures a realistic evaluation framework.

Standardization was applied using the StandardScaler technique to normalize the features:

$$x'_i = \frac{Xi - \mu}{\sigma}$$

where (x_i) is the original feature value, (μ) is the mean of the training data, and (σ) is the standard deviation.

5.4 Data Balancing

The original dataset exhibited a significant class imbalance across the intrusion categories. For example, the Normal class contained 50,000 instances (23.69%), whereas the MITM (Man-in-the-Middle**)** attack class contained only 1,043 instances (0.49%). Such class imbalance can bias the learning process toward majority classes, resulting in poor classification performance on minority attack categories.

To address this challenge, the Synthetic Minority Oversampling Technique (SMOTE) was employed. SMOTE generates synthetic samples for minority classes by interpolating between existing minority observations and their nearest neighbors, thereby improving class representation without simply duplicating existing samples.

A synthetic sample is generated according to:

$$x_{new} = x_i + \lambda(x_{zi} - x_i)$$

where (x_i) is a minority class sample, (x_{zi}) is one of its nearest neighbors, and ($\lambda \in [0,1]$) is a random interpolation factor, and x_{new} denotes the generated synthetic sample.

To prevent information leakage and ensure a realistic evaluation framework, the dataset was first partitioned into training and testing subsets using a stratified train-test split. Feature standardization was subsequently performed using parameters derived exclusively from the training dataset. Thereafter, SMOTE was applied only to the standardized training dataset, while the testing dataset remained completely unchanged and was used solely for model evaluation. This procedure ensured that no information from the testing data influenced either the scaling or oversampling processes, thereby providing a more reliable assessment of model generalization performance.

5.5 System model for training and testing deep learning models

The proposed framework for network intrusion detection and multiclass attack classification is illustrated in Figure 4. The preprocessed UNSW-NB15 dataset was used to develop and evaluate the ANN, CNN, and RNN-LSTM models. Model performance was assessed using multiple evaluation metrics, including accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrices.

The ANN, CNN, and RNN-LSTM architectures were selected because they represent three widely adopted deep learning paradigms commonly used in intrusion detection research. The ANN model provides a baseline fully connected nonlinear classifier, the CNN architecture supports hierarchical feature extraction through convolutional operations, and the RNN-LSTM architecture enables gated representation learning across structured input representations.

The comparative evaluation of these architectures under identical preprocessing, balancing, and evaluation conditions was intended to provide insight into their relative strengths and limitations for multiclass intrusion detection using the UNSW-NB15 dataset.

Prior to model development, the dataset underwent a systematic preprocessing pipeline consisting of data cleaning, feature selection, dataset partitioning, feature standardization, and class balancing. First, irrelevant and redundant attributes were removed through feature selection. The dataset was then partitioned into training and testing subsets using a stratified train-test split. Feature standardization was subsequently performed using parameters derived exclusively from the training dataset. Thereafter, the Synthetic Minority Oversampling Technique (SMOTE) was applied only to the standardized training dataset to address class imbalance, while the testing dataset remained unchanged.

The processed training data were subsequently used to train the ANN, CNN, and RNN-LSTM models, whereas the testing dataset was used exclusively for performance evaluation. This workflow ensures methodological consistency, prevents information leakage, and provides a reliable assessment of model generalization capability.

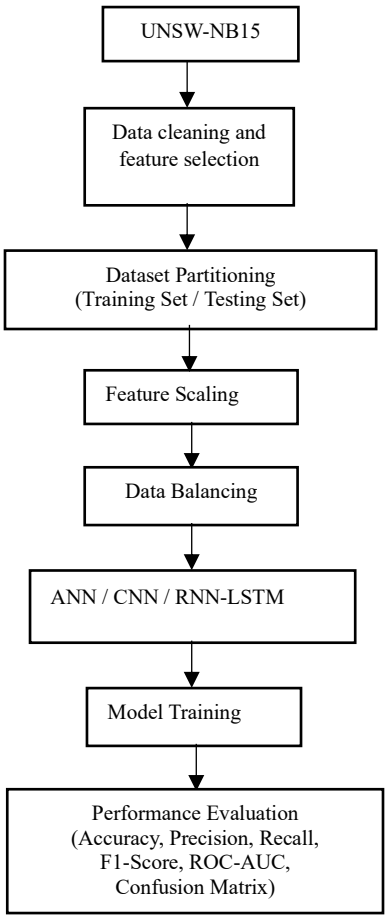


Figure 4. System model for training and testing of deep learning model

6.0 Summary of proposed ANN Model Results

The performance of the ANN model is summarized in Table 4. The model achieved an overall accuracy of 91.58%, with precision, recall, and F1-score values of 92.11%, 91.58%, and 91.65%, respectively. These results indicate that the model was able to effectively distinguish between normal network traffic and multiple intrusion categories within the UNSW-NB15 dataset.

The close agreement among the accuracy, precision, recall, and F1-score values suggests that the ANN model maintained a balanced classification performance across the evaluated classes. Analysis of the class-specific metrics revealed strong detection performance for several attack categories, including Backdoor, DoS, Ransomware, and Scanning attacks. However, relatively lower performance was observed for classes such as MITM (Man-in-the-Middle) and XSS (Cross-Site Scripting), indicating that these attack categories remain more challenging to classify accurately.

The results demonstrate the capability of the ANN architecture to learn meaningful patterns from network traffic data and provide reliable multiclass intrusion classification. Nevertheless, variations in performance across attack categories highlight the importance of comparing ANN with more advanced deep learning architectures such as CNN and RNN-LSTM to determine whether improved feature learning can enhance minority-class detection performance.

Table 4: Summary of ANN Model Performance

MODEL	Accuracy	Precision	Recall	F1-Score
ANN	0.9158	0.9211	0.9158	0.9165

6.1 Summary of proposed CNN Model Results

The performance of the CNN model is summarized in Table 5. The model achieved an overall accuracy of 94.64%, with precision, recall, and F1-score values of 95.10%, 94.64%, and 94.71%, respectively. These results indicate strong classification capability and demonstrate an improvement over the performance achieved by the ANN model.

The close agreement among the accuracy, precision, recall, and F1-score values suggests that the CNN model maintained a balanced classification performance across the evaluated intrusion categories. Analysis of the class-specific metrics revealed strong detection performance for attack classes such as DoS, Normal, Password, and Ransomware, which achieved high precision and recall values. The Backdoor class achieved perfect recall but comparatively lower precision, indicating that although most Backdoor attacks were correctly detected, a number of false positive classifications were also produced.

The confusion matrix and ROC-AUC analysis further indicate that certain attack categories, particularly DDoS and MITM, remain more challenging to distinguish from related attack behaviors. Nevertheless, the overall performance demonstrates the effectiveness of the CNN architecture in automatically learning discriminative feature representations from network traffic data.

Compared with the ANN model, the CNN architecture achieved higher overall classification performance across all major evaluation metrics, suggesting that convolutional feature extraction contributed positively to multiclass intrusion detection within the UNSW-NB15 dataset.

Table 5: Summary of CNN Model Performance

MODEL	Accuracy	Precision	Recall	F1-Score
CNN	0.9464	0.9510	0.9464	0.9471

6.2 Training and Validation Results of RNN-LSTM Model

The training and validation performance of the proposed RNN-LSTM model is presented in Figure 5. The model was trained for 100 epochs, during which the learning curves demonstrated progressive improvement in classification performance and convergence behavior.

As shown in Figure 5, training and validation accuracy increased steadily throughout the training process, while the corresponding loss values decreased consistently. The model achieved an initial training accuracy of 67.20% and validation accuracy of 81.26% during the early training stage. As training progressed, both metrics improved substantially, exceeding 95% accuracy and indicating effective learning of the underlying network traffic patterns.

Similarly, the precision and recall curves exhibited continuous improvement across successive epochs. Training precision increased from 79.98% to approximately 96%, while recall improved from 55.17% to approximately 95%. The corresponding validation precision and recall values followed closely related trends, suggesting that the model generalized effectively to unseen data.

Furthermore, the relatively small gap observed between the training and validation curves for loss, accuracy, precision, and recall indicates stable learning behavior and limited evidence of severe overfitting. The consistent convergence of both training and validation metrics demonstrates the ability of the RNN-LSTM architecture to learn discriminative feature representations for multiclass intrusion classification using the UNSW-NB15 dataset.

Overall, the learning curves confirm that the model achieved stable convergence during training and provide evidence of its suitability for subsequent intrusion detection and attack classification tasks.

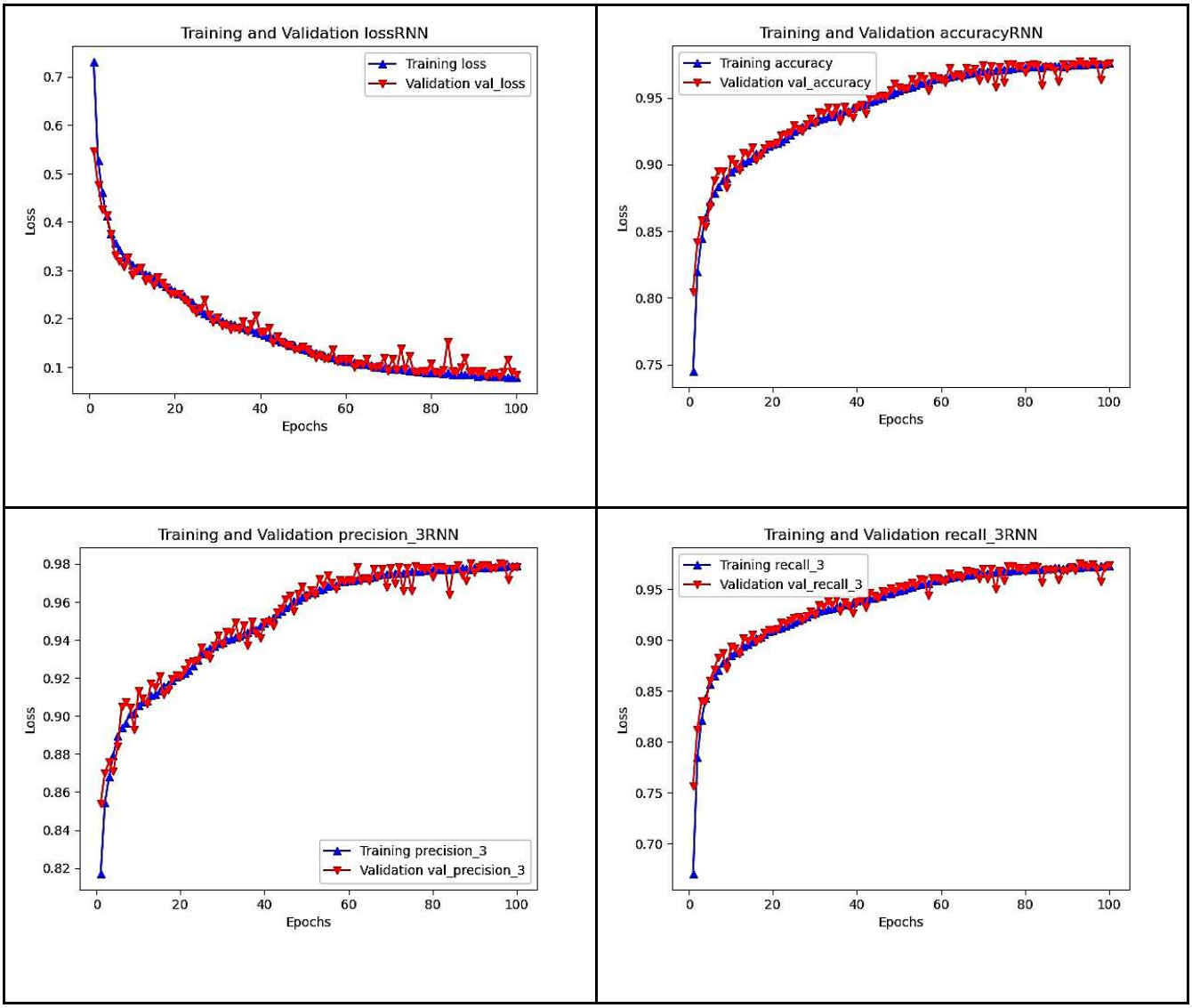


Figure 5. Training and validation results of RNN-LSTM

6.3 Confusion Matrix Analysis of the RNN-LSTM Model

The confusion matrix presented in Figure 6 provides a detailed view of the classification performance of the RNN-LSTM model across the intrusion categories. Correct classifications are represented along the diagonal of the matrix, while off-diagonal entries indicate misclassified instances.

As shown in Figure 6, the model achieved strong classification performance for several attack categories, including Backdoor, XSS, Ransomware, and Normal traffic, which recorded high numbers of correctly classified instances and relatively few classification errors. This indicates that the RNN-LSTM architecture was able to learn distinctive feature representations for these categories.

However, the confusion matrix also reveals several recurring misclassification patterns. In particular, MITM attacks were frequently confused with Backdoor and Injection attacks, while DDoS attacks occasionally overlapped with Injection and Scanning categories. Similarly, Scanning and Password attacks exhibited moderate levels of mutual misclassification. These observations suggest that certain attack classes share similar traffic characteristics, making them more difficult to distinguish accurately.

Despite these challenges, the majority of predictions remained concentrated along the diagonal of the confusion matrix, indicating a high overall classification accuracy. The relatively limited number of off-diagonal errors demonstrates that the RNN-LSTM model was effective in learning discriminative patterns from the UNSW-NB15 dataset and was capable of performing reliable multiclass intrusion classification.

The confusion matrix analysis further highlights the importance of evaluating class-specific behavior in addition to aggregate performance metrics, as certain attack categories remain inherently more challenging to detect than others. These findings provide valuable insights into the strengths and limitations of the RNN-LSTM architecture for network intrusion detection applications.

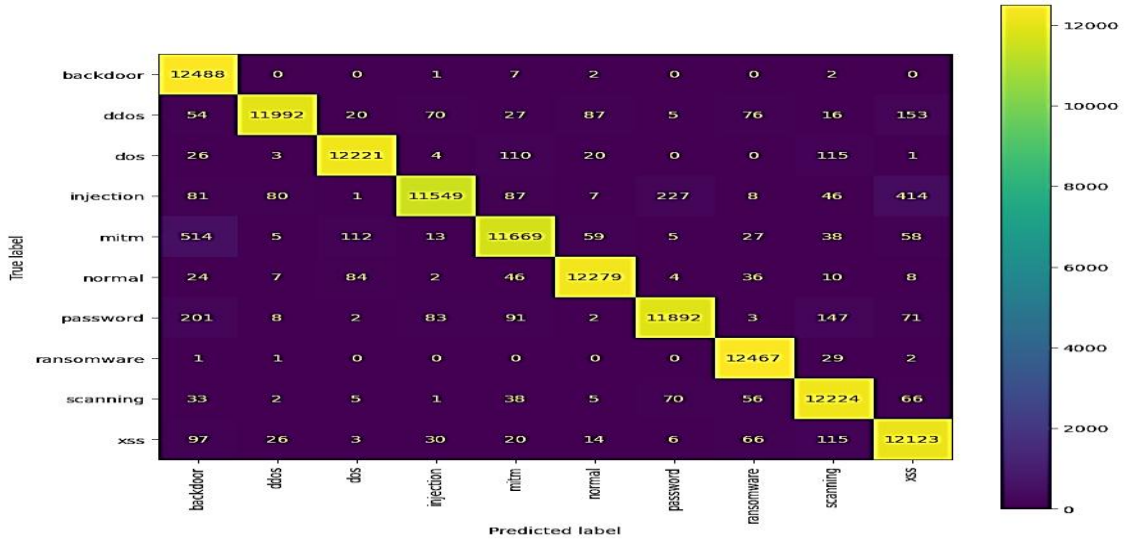


Figure 6: Confusion Matrix of RNN-LSTM model.

6.4 ROC-AUC Analysis of the RNN-LSTM Model

The Receiver Operating Characteristic (ROC) curves presented in Figure 7 provide an assessment of the discriminative capability of the RNN-LSTM model across the intrusion categories. The Area Under the ROC Curve (AUC) values were consistently high for all classes, indicating strong class-separation performance and effective identification of both normal and malicious network traffic patterns.

As shown in Figure 7, the Backdoor and Ransomware classes achieved the highest AUC values of 0.9999 and 0.9998, respectively, while the DDoS class recorded the lowest AUC value of 0.9985. Despite being the lowest among the evaluated categories, the DDoS class still demonstrated a very high level of separability. The remaining attack categories similarly achieved AUC values exceeding 0.998, indicating consistent classification performance across the multiclass intrusion detection task.

The overall AUROC value of 0.9991 further demonstrates the strong discriminative capability of the RNN-LSTM model. These results suggest that the model was able to effectively distinguish among the intrusion categories represented in the UNSW-NB15 dataset while maintaining low false-positive and false-negative rates.

Although the ROC-AUC results indicate excellent classification performance, the confusion matrix analysis revealed that certain attack categories, particularly MITM, DDoS, and Injection attacks, still exhibited some degree of overlap and misclassification. Consequently, ROC-AUC results should be interpreted alongside class-specific performance metrics and confusion matrix analysis to provide a comprehensive assessment of model effectiveness.

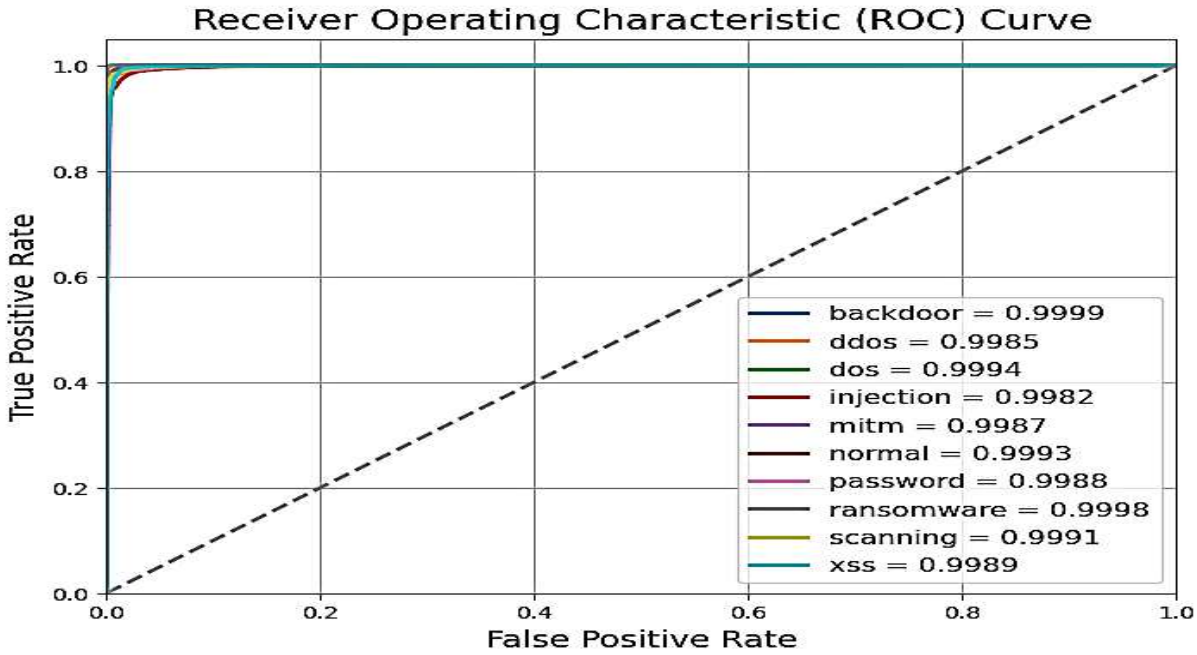


Figure 7: ROC Curve of RNN-LSTM

6.5 Summary of Proposed RNN-LSTM Model Results

The performance of the RNN-LSTM model is summarized in Table 6. The model achieved an overall accuracy of 97.42%, with precision, recall, and F1-score values of 97.45%, 97.42%, and 97.42%, respectively. These results represent the highest classification performance among the evaluated deep learning architectures and demonstrate the effectiveness of the RNN-LSTM model for multiclass intrusion detection using the UNSW-NB15 dataset.

The close agreement among the accuracy, precision, recall, and F1-score values indicates a balanced classification performance across the intrusion categories. Class-specific analysis revealed strong detection capability for Backdoor, DoS, Normal, Password, Ransomware, and Scanning attacks, which achieved F1-scores approaching or exceeding 0.98. In contrast, DDoS and MITM attacks exhibited slightly lower precision and recall values, indicating that these attack categories remained comparatively more challenging to classify accurately.

The confusion matrix and ROC-AUC analyses further support these findings by revealing occasional overlap between certain attack categories while maintaining strong overall class separability. Despite these challenges, the model consistently achieved high detection rates across all intrusion classes.

The training loss value of 0.1043 and the observed convergence behavior indicate that the model successfully learned discriminative feature representations from the network traffic data. Although the RNN-LSTM model required a longer training time than the ANN and CNN architectures, its superior classification performance demonstrates the effectiveness of memory-based feature learning for intrusion detection and multiclass attack classification.

Table 6: Summary of RNN-LSTM Model Performance

Model	Accuracy	Precision	Loss	Recall	F1-Score
RNN-LSTM	0.9742	0.9745	0.1043	0.9742	0.9742

6.6 Comparative Analysis of the proposed Models Using the UNSW-NB15 Dataset

The comparative performance of the ANN, CNN, and RNN-LSTM models is presented in Table 7. The results demonstrate that the RNN-LSTM model achieved the highest classification performance across all evaluation metrics, recording an accuracy of 97.42%, precision of 97.45%, recall of 97.42%, and F1-score of 97.42%. These results indicate that the memory-based learning mechanism of the RNN-LSTM architecture was particularly effective in capturing complex feature relationships within the intrusion dataset.

The CNN model achieved the second-best performance, with an accuracy of 94.64% and an F1-score of 94.71%. This suggests that convolutional feature extraction contributed positively to intrusion classification, although its performance remained below that of the RNN-LSTM architecture. The ANN model recorded the lowest overall performance, achieving an accuracy of 91.58% and an F1-score of 91.65%, indicating comparatively weaker feature-learning capability for the evaluated intrusion detection task.

From a computational perspective, the ANN model required the shortest training time (220.25 s), followed by the CNN model (491.52 s), while the RNN-LSTM model required the longest training time (1698.24 s). This observation highlights an important performance-complexity tradeoff: although the RNN-LSTM architecture achieved the highest predictive performance, it incurred substantially higher computational cost. Conversely, the ANN model offered the fastest training time but at the expense of lower classification accuracy.

Overall, the results indicate that the RNN-LSTM model provided the best classification performance for multiclass intrusion detection using the UNSW-NB15 dataset, whereas the CNN model offered a balance between predictive performance and computational efficiency. The ANN model demonstrated the lowest computational cost but also the lowest detection performance among the evaluated architectures.

Table 7: Comparative Analysis of the proposed Models with UNSW-NB15

Model	Accuracy	Precision	Recall	F1-Score	Time (Secs)
ANN	0.9158	0.9211	0.9158	0.9165	220.25
CNN	0.9464	0.9510	0.9464	0.9471	491.52
RNN-LSTM	0.9742	0.9745	0.9742	0.9742	1698.24

7. Conclusion and Recommendations

This study presented a comparative evaluation of three deep learning architectures—Artificial Neural Network (ANN), Convolutional Neural Network (CNN), and Recurrent Neural Network with Long Short-Term Memory (RNN-LSTM)—for network intrusion detection and multiclass attack classification using the UNSW-NB15 dataset. A systematic preprocessing framework involving feature selection, dataset partitioning, feature standardization, and class balancing using the Synthetic Minority Oversampling Technique (SMOTE) was implemented to support effective model training and evaluation.

Experimental results demonstrated that all three deep learning architectures were capable of identifying normal and malicious network traffic with high classification accuracy. Among the evaluated models, the RNN-LSTM architecture achieved the best overall performance, recording an accuracy of 97.42%, precision of 97.45%, recall of 97.42%, and F1-score of 97.42%. The CNN model achieved the second-best performance, while the ANN model recorded the lowest classification accuracy among the evaluated architectures. The comparative analysis further revealed that although the RNN-LSTM model required a longer training time, it provided superior classification capability across the intrusion categories.

The findings demonstrate the effectiveness of deep learning techniques for intrusion detection and highlight the advantages of memory-based learning architectures for capturing complex relationships within network traffic data. The study also emphasizes the importance of appropriate preprocessing, feature representation, and class balancing in improving intrusion detection performance.

Future research should investigate the application of the proposed framework to additional intrusion detection datasets and evaluate model performance under real-time network traffic conditions. Furthermore, emerging architectures such as transformer-based models, federated learning frameworks, and explainable artificial intelligence techniques may be explored to

improve detection capability, interpretability, and deployment readiness in practical cybersecurity environments.

References

[1] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges,” Cybersecurity, vol. 2, no. 1, pp. 1–22, 2019.

[2] W. Stallings and L. Brown, Computer Security: Principles and Practice, 4th ed. Pearson, 2018.

[3] E. Cole, Advanced Persistent Threat: Understanding the Danger and How to Protect Your Organization. Elsevier, 2012.

[4] S. Noel and S. Jajodia, “Managing attack graph complexity through visual hierarchical aggregation,” in Proc. ACM Workshop on Visualization and Data Mining for Computer Security, 2004, pp. 109–118.

[5] D. E. Denning, “An intrusion-detection model,” IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222–232, 1987.

[6] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in Proc. IEEE Symposium on Security and Privacy, 2010, pp. 305–316.

[7] S. Axelsson, “Intrusion detection systems: A survey and taxonomy,” Technical Report No. 99-15, Chalmers University of Technology, Sweden, 2000.

[8] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[10] N. Moustafa and J. Slay, “UNSW-NB15: A comprehensive data set for network intrusion detection systems,” in Proc. Military Communications and Information Systems Conference (MilCIS), 2015, pp. 1–6.

[11] H. Hindy et al., “A taxonomy of network threats and the effect of current datasets on intrusion detection systems,” IEEE Access, vol. 8, pp. 104650–104675, 2020.

[12] M. F. Elrawy, A. I. Awad, and H. F. A. Hamed, “Intrusion detection systems for IoT-based smart environments: A survey,” Journal of Cloud Computing, vol. 7, no. 1, Art. no. 21, 2018, doi: 10.1186/s13677-018-0123-6.

[13] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, “A deep learning approach for network intrusion detection system,” in Proc. EAI International Conference on Bio-inspired Information and Communications Technologies (BICT), New York, NY, USA, 2016, pp. 21–26.

[14] M. A. Ferrag, L. Maglaras, H. Janicke, J. Jiang, and S. Moschoyiannis, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,” Journal of Information Security and Applications, vol. 50, p. 102419, 2020.

[15] A. Verma and V. Ranga, “Machine learning based intrusion detection systems for IoT applications,” Wireless Personal Communications, vol. 111, no. 4, pp. 2287–2310, 2020.

[16] H. He and E. A. Garcia, “Learning from imbalanced data,” IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 9, pp. 1263–1284, 2009.

[17] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” Intelligent Data Analysis, vol. 6, no. 5, pp. 429–449, 2002.

[18] Z. Dai, L. Y. Por, Y.-L. Chen, J. Yang, C.-S. Ku, et al., “An intrusion detection model to detect zero-day attacks in unseen data using machine learning,” PLOS ONE, vol. 19, no. 9, pp. 1–25, 2024.

[19] L. Sana, M. M. Nazir, J. Yang, L. Hussain, Y.-L. Chen, C.-S. Ku, M. Alatiyyah, S. A. Alateyah, and L. Y. Por, “Securing the IoT Cyber Environment: Enhancing Intrusion Anomaly Detection With Vision Transformers,” IEEE Access, vol. 12, pp. 82443–82468, 2024.

[20] J. Yang, Y. Wu, Y. Yuan, H. Xue, S. Bourouis, M. Abdel-Salam, and L. Y. Por, “LLM-AE-MP: Web Attack Detection Using a Large Language Model with Autoencoder and Multilayer Perceptron,” Expert Systems with Applications, vol. 274, p. 126982, 2025.

[21] 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 dataset,” EURASIP Journal on Wireless Communications and Networking, vol. 2021, no. 1, pp. 1–23, 2021.

[22] N. Moustafa and J. Slay, “The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 dataset and the comparison with the KDD99 dataset,” Information Security Journal: A Global Perspective, vol. 25, nos. 1–3, pp. 18–31, 2016.

[23] M. Dwibedi, M. Pujari, and W. Sun, “A comparative study on contemporary intrusion detection datasets for machine learning research,” in Proc. IEEE International Conference on Electro Information Technology (EIT), 2020, pp. 242–247.

[24] 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, 2014.

[25] L. Y. Por, Z. Dai, S. J. Leem, Y. Chen, J. Yang, F. Binbeshr, K. Y. Phan, and C. S. Ku, “A Systematic Literature Review on AI-Based Methods and Challenges in Detecting Zero-Day Attacks,” IEEE Access, vol. 12, pp. 144150–144163, 2024, doi: 10.1109/ACCESS.2024.3455410.

[26] F. Ullah, S. Ullah, G. Srivastava and J. Lin: IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic. Digit. Commun. Networks 10(1): 190-204 (2024)

[27] C. Xi, H. Wang, and X. Wang, "A novel multi-scale network intrusion detection model with transformer," Scientific Reports, vol. 14, Art. no. 23239, 2024, doi: 10.1038/s41598-024-74214-w.

[28] H. Zhou, H. Zou, W. Li, D. Li, Y. Kuang, HiViT-IDS: "An Efficient Network Intrusion Detection Method Based on Vision Transformer. *Sensors"*. 2025; 25(6):1752. https://doi.org/10.3390/s25061752

[29] X. Chen et al., "Overcoming Class Imbalance in Network Intrusion Detection," in Proc. ACM International Conference on Information Technology, 2024.

[30] V. Shanmugam, R. Razavi-Far, E. Hallaji, "Addressing Class Imbalance in Intrusion Detection: A Comprehensive Evaluation of Machine Learning Approaches". *Electronics*. 2025; 14(1):69. https://doi.org/10.3390/electronics14010069

[31] M. B. Musthafa, S. Huda, Y. Kodera, M. A. Ali, S. Araki, J. Mwaura, and Y. Nogami, "Optimizing IoT Intrusion Detection Using Balanced Class Distribution, Feature Selection, and Ensemble Machine Learning Techniques," Sensors, vol. 24, no. 13, Art. no. 4293, 2024, doi: 10.3390/s24134293.

[32] 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.

[33] N. AL-Nomasy, A. Alamri, A. Aljuhani, and P. Kumar, "Transformer-based knowledge distillation for explainable intrusion detection system," Computers & Security, vol. 154, Art. no. 104417, 2025, doi: 10.1016/j.cose.2025.104417.

[34] I. Koukoulis, I. Syrigos, and T. Korakis, "Self-Supervised Transformer-based Contrastive Learning for Intrusion Detection Systems," ArXiv abs/2505.08816, 2025.

[35] M. Abd Elaziz, I. A. Fares, A. Dahou, and M. Shrahili, "Federated learning framework for IoT intrusion detection using tab transformer and nature-inspired hyperparameter optimization," Frontiers in Big Data, vol. 8, Art. no. 1526480, 2025, doi: 10.3389/fdata.2025.1526480.

[36] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[37] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[38] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

Acknowledgements

We thank all the reviewers for their effort, time and consideration in reviewing our article which has help in quality improvement

Funding

There was no funding to complete this research.

Author contributions

Conceptualisation, AE, PA, SO. Investigation and methodology, AE, SO, BS. Project administration, PA, AE. Resources, AE, PA SO, BS. Supervision, PA, SO. BS. Writing of the original draft, AE. Writing of the review and editing, AE, EJ, PA. Software AE, PA, BS. Validation, AE, SO, PA. Formal analysis, AE, PA. Data curation, AE, EJ. Visualization, AE, PA.

Correspondence

Corresponding author: Akpasam Ekanem

Corresponding author's email: **akpasamekanem@aksu.edu.ng**