Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 3, 2026

A ROBUST MACHINE LEARNING FRAMEWORK FOR ANOMALY-BASED INTRUSION DETECTION IN SOFTWARE-DEFINED NETWORKS

Muhammad Hanif Hussain Khan[*1], Arsalan Khan[2], Saif Ullah Noor[3], Wasiat Khan[4], Amjad Khan[5]

[*1]*Department of Computer Science, Brains Institute Peshawar, Khyber Pakhtunkhwa, Pakistan*
[2,3,2]*Department of Computer Science, University of Science and Technology Bannu, Khyber Pakhtunkhwa, Pakistan*
[4]*Department of Software Engineering, University of Science and Technology Bannu, Khyber Pakhtunkhwa, Pakistan*

DOI: https://doi.org/10.5281/zenodo.19199992

Keywords
Intrusion Detection System, Software Defined Networking, Deep Learning, ANN, Ensemble Machine Learning, Anomaly Detection

Article History
Received: 25 January 2026
Accepted: 10 March 2026
Published: 24 March 2026

Corresponding Author: *
Muhammad Hanif Hussain Khan

Abstract
Software Defined Networking (SDN) enables centralized network management, enhanced programmability, and improved flexibility by decoupling the control plane from the data plane. However, this architecture introduces security vulnerabilities that challenge conventional signature-based Intrusion Detection Systems (IDS), which are often ineffective against evolving or novel cyber threats. This study investigates the efficiency of ensemble machine learning and deep learning models for anomaly-based IDS within SDN environments. The CIC-IDS-2017 benchmark dataset was employed for evaluation due to its realistic traffic patterns and inclusion of modern attack scenarios. A multi-stage preprocessing pipeline was applied, including Min-Max normalization, label encoding and the Synthetic Minority Over-sampling Technique (SMOTE) to address severe class imbalance in network traffic data. Several machine learning techniques, including XGBoost, Random Forest (RF), and Support Vector Machine (SVM), were examined alongside a lightweight custom-designed Artificial Neural Network (ANN). The suggested ANN contains five layers, an input layer, three hidden layers, and one output layer, designed to capture complex nonlinear patterns in network traffic. Experimental results show that the ANN achieved superior performance with a peak accuracy of 99.80%, while among machine learning models; RF outperformed all others with an accuracy of 99.05%. These findings indicate that deep learning-based approaches are highly effective in detecting diverse intrusion types, capturing complex traffic patterns, and maintaining low false negative and false positive rates, highlighting their suitability for SDN security.

1. INTRODUCTION
Software-Defined Networking (SDN) is a design paradigm revolution in the current network architecture, which has radically altered the manner in which networks are designed, operated, and controlled. Contrary to other traditional networking infrastructures, where the control plane and the data plane overlap each other inside individual network machines, including routers and switches, SDN isolates the two planes to make centralized network control and increased programmability possible [1]. Such architectural decoupling allows network administrators to dynamically define network policy, free up

resource usage, and enact real-time changes in the network via a centralized SDN controller. As a result, SDN can improve network flexibility, scalability, and overall efficiency [2]. Besides several advantages over traditional architectures, SDN's centralized architecture presents a number of security issues. The SDN controller is the main component of the network infrastructure that controls the whole network infrastructure and is thus a prime target of cyber-attacks. A successful attack on the controller can disrupt network services, enable unauthorized access, and lead to large-scale data leakage. Therefore, an effective security apparatus of an SDN setting has turned into a crucial study problem of contemporary network administration [3].

The role of intrusion detection systems (IDS) in securing network infrastructures is critical in that they detect malicious activities and unauthorized access to networks. Conventional IDS models, especially signature-based models, use established attack patterns in order to identify intrusions. These systems are, however, not always able to identify new attacks that arise or are yet unknown to them, which are becoming more prevalent in dynamic and programmable networks like SDNs. To overcome this shortcoming, intrusion detection methods based on anomalies have become a major concern. Such methods provide a point of reference of normal network behavior and identify anomalies that can be related to malicious activities [4-5]. The recent developments in ML and DL have also increased the performance of the anomaly-based IDS. RF, SVM, and XGBoost machine learning algorithms have a good record of detecting complex network traffic patterns. Moreover, deep learning models like ANNs have strong feature learning functions that offer a greater classification of normal and malicious traffic. Another method that can be employed to improve the performance of identification is through ensemble learning procedures that combine two or more models, exploiting the strengths of every algorithm [6].

In this work, a smart anomaly-based Network Intrusion Detection System (NIDS) is designed in SDN settings through both ML and DL methods. The proposed framework uses the CIC-IDS-2017 dataset, which has real network traffic and recent attack situations. The data preprocessing pipeline is used carefully with label encoding, min-max normalization, and SMOTE to overcome the problems of data imbalance [7]. Several machine learning models are used and compared with the outcomes of a lightweight ANN model to examine how useful they are in terms of classifying network intrusions, which include RF, XGBoost, and SVMs. The experimental findings indicate that the DL model performed better in the detection of malicious activities with low false negative and false positive rates. The results indicate that the use of intelligent learning-based methods is effective to enhance network security in dynamic SDN environments. The most important findings of this study can be presented as follows:

- An anomaly-based intrusion detection framework is suggested for SDN using ML and DL techniques to improve the identification of modern network attacks.
- A comprehensive data preprocessing techniques were used to enhance model performance.
- A comparative evaluation of multiple machine learning algorithms and a lightweight ANN was conducted using the CIC-IDS-2017 training data.
- Experimental outcomes demonstrate that the proposed ANN model achieved comparatively good detection score while maintaining low false negative and false positive rates.

2. RELATED WORK

IDS are security tools based on hardware or software used to monitor and analyze data that happens within a computer system or a computer network and determine indicators of causes of a security violation. These breaches can be seen in terms of violations of the laid-down security policies, unauthorized activities, or adhering to normal security practices and acceptable usage guidelines. IDS are important in contemporary network security architecture because it is utilized to monitor network traffic and system activities to identify the presence of suspicious activity. In this aspect, IDS can be described as a watchful

surveillance organization, which assists in detecting and addressing a possible security threat and, hence, improves the overall security of network infrastructures [8]. Utilizing the famous networks attacks data called UNSW-NB15 for model training and assessment, Moualla et al. [9] developed a novel network IDS approach that is vital to online security and fights against attacks on the internet. It was a multiphase, dynamically expandable multilevel machine learning network. The SMOTE approach was utilized to tackle the unbalanced followed by the ET classification method based on the Gini imperfection criteria and, finally, a previously trained ELM to categorize each assault using a binary classification. A logistic regression (LR) structure was employed to produce soft judgments for all categories, utilizing the ELM classification algorithm's results as input to the dense layer, achieving a 98.43% success score.

Nimbalkar and Kshirsagar [10] used the KDD Cup 1999 Dataset and Bot-IoT Dataset to test the effectiveness of their feature engineering method for IDS applied to Gain Ratio (GR) and Information Gain (IG). The authors have chosen the top 50% of significant features in their study to develop a model to help in the identification of Denial of Service (DoS) as well as Distributed Denial of Service (DDoS) threats. In particular, 16 features were picked out of the Bot-IoT data and 19 key attributes out of the KDD Cup 1999 network attacks data. The algorithm was subsequently trained and tested on the attributes selected from both the datasets with the JRip classifier so as to obtain optimal detection results. The experimental results obtained high classification score as it got 99.99% on the Bot-IoT data and 99.57% on the KDD Cup 1999 data.

Kumar et al. [11] put forward misuse-based IDS, which is intended to identify malicious activities and guard against a number of contemporary online attacks, such as DoS attacks, exploits, probes, and generic attacks. The proposed model has undergone the evaluation of its performance by the well-known dataset called UNSW-NB15, in which the most important performance metrics were the False Alarm Rate (FAR) and Intrusion Detection Rate (IDR). In this case, the authors used the Information Gain (IG) to single out the most important features, and the initial 47 features were narrowed down to 13 important features. The classifier task was then utilized with the C5.0 classifier that obtained an accuracy of 99.37% in detection, proving the efficiency of the suggested IDS model.

Ahmad et al. [12] have suggested a novel ML-based intrusion detection approach, which used a feature clustering strategy to improve detection performance. Features were organized into distinct groups such as flow, MQTT, and TCP attributes features in their approach. The common pitfalls that were helped to be overcome by this grouping mechanism were those related to over fitting caused by high data dimensionality and uneven distribution of classes in the data. In order to test the usefulness of the clustered feature groups, the authors used several supervised ML techniques, namely, RF and SVM. The UNSW-NB15 dataset was utilized to conduct the experiments both in training and in testing. The findings have shown that the Random Forest classifier gave a good result with 97.90% detection score in binary identification and 96.99% accuracy in multi-category identification, which illustrates the suitability of the suggested clustering-based IDS framework. The study in [13] offered the way of dimensionality reduction of the dataset. In order to enhance efficiency even more in the situation of processing large-sized data, a MapReduce method was applied at the same time to split the input data and determine the most meaningful features. The processed data was then classified into normal or attack with the help of the RF model after the feature engineering stage. The effectiveness of the suggested model was tested on the well-known KDD Cup 1999 dataset, which includes data about normal and abnormal network activities. Out of the initial dataset, 15 important features were obtained and utilized to train and test the algorithm. The evaluation outcomes depict that the suggested method had an overall classification accuracy of 93.9%.

Talita et al. [14] suggested using a hybrid algorithm that was a combination of an attribute chosen algorithm and Particle Swarm Optimization (PSO) algorithm used for feature set reduction followed by the Naive Bayes (NB)

identification technique. The algorithm was tested on a public dataset, consisting of over 400000 records and upwards of 40 network traffic aspects. In order to enhance the efficacy of the computations and minimize the usage of memory, PSO was applied to select the most relevant attributes, with 38 important features being selected out of the initial array of features. The Naive Bayes intrusion detector was then trained using the chosen features. The evaluation outcomes exhibited that the suggested method had a detection rate of 99.12% as well as increased processing efficiency and minimized processing time relative to some of the traditional feature selection strategies. Seth et al. [15] proposed an intrusion detection approach, which minimized the prediction latency by minimizing the complexity of a model using a hybrid feature selection (HFS) approach. LightGBM was used to develop the classification model because it is a rapid and effective gradient boosting model that is characterized by high performance during large-scale machine learning projects. The suggested method was assessed by the use of the CIC-IDS2018 dataset. The approach optimally decreased the computational time and prediction latency by combining a hybrid feature selection with the LightGBM classifier. The experimental results showed that prediction latency has dropped to 2.25% with a reduction of 44.52%, and the model development time dropped to 17.94% with a reduction of 52.68%. Moreover, the model generated high detection score, with an accuracy of 98.05%, recall of 96.25%, and precision of 98.99% and relatively reduced identification time. The outcomes show the usefulness of the functionality of the feature selection methods and LightGBM algorithm with regard to effective and accurate intrusion detection.

Stiawan et al. [16] provided a model of analyzing big data of network traffic to enhance the efficacy and identification score of anomaly detection. Instead, they aimed at defining the most important and relevant features in large network data to enhance the performance of detectors and minimize processing time. The dataset used in the study was the CIC-IDS2017, in which Information Gain (IG) was used to choose significant attributes. The features were then grouped and sorted based on their lowest weight after feature selection. Several machine learning classifiers were then trained on the processed dataset, including KNN, SVM, Random Tree (RT), RF, NB, BayesNet (BN), and J48. It has been found that the number of features chosen using IG had a considerable effect on the accuracy of classification and time of execution. The best-balanced results were received with the Random Forest classifier, which was trained on 22 selected features and got an accuracy of 99.86%. However, the J48 classifier was slightly more accurate with 99.87% and used 52 selected features but took a longer training time. These results show the importance of feature engineering in improving the strength of IDS in case of large network data.

In conclusion, the related work shows that ML and DL models can be applied to enhance the strength of IDS by maximizing the detection rate of cyber-attacks and decreasing computation time. Different approaches, such as feature selection, clustering, and hybrid optimization approaches, have been popularly used to process the high-dimensional network data. The well-known classifiers like the SVM, RF, Naive Bayes, and gradient boosting techniques have demonstrated good strength in the detection of various network attacks. Along with these developments, there are challenges in dataset imbalance, feature redundancy, real-time detection performance, and other problems that need to be further explored, which encourages the study for more robust and scalable models of IDS.

3. PROPOSED METHOD

This section contains the overall workflow of the study, including the training dataset, data preprocessing techniques, models used and evaluation results. The architectural design contains both the ML models and a custom ANN. The general flow of the study is depicted in Figure 1 below.

3.1 Training Dataset

The proposed study used a famous training dataset known as the CIC-IDS-2017 for algorithm training and evaluation. It has a total of 2830743

network traffic records collected over five days. Each record contains 78 numerical features that were derived using the CICFlowMeter tool and which describe the essential flow statistics, including length of a packet, data flow duration and inter-arrival times of data packets. The training data was divided into train set and validation set with the ratio of 0.80 and 0.20, respectively. The dataset contains 1,818,478 normal traffic records and 446,117 attack traffic records for training, and 454,619 normal records and 111,529 attack records are used for validation. The attack traffic includes several types, such as DDoS attacks (HOIC, LOIC-HTTP, SlowHTTPTest, Hulk, GoldenEye, Slowloris, LOIC-UDP), botnet activity, FTP and SSH brute-force attacks, infiltration attempts and web-based threats (XSS, Brute Force, and SQL Injection). Yet, within the context of this study, all types of attacks have been used as a single class, and this approach allows the classification framework to be described as binary with the ability to differentiate among normal traffic and multiple types of malicious attacks. The dataset is publicly accessible and can be easily obtained from the Kaggle platform [17]. Figure 2 demonstrates the class distribution of benign traffic and various attack types in the training data.

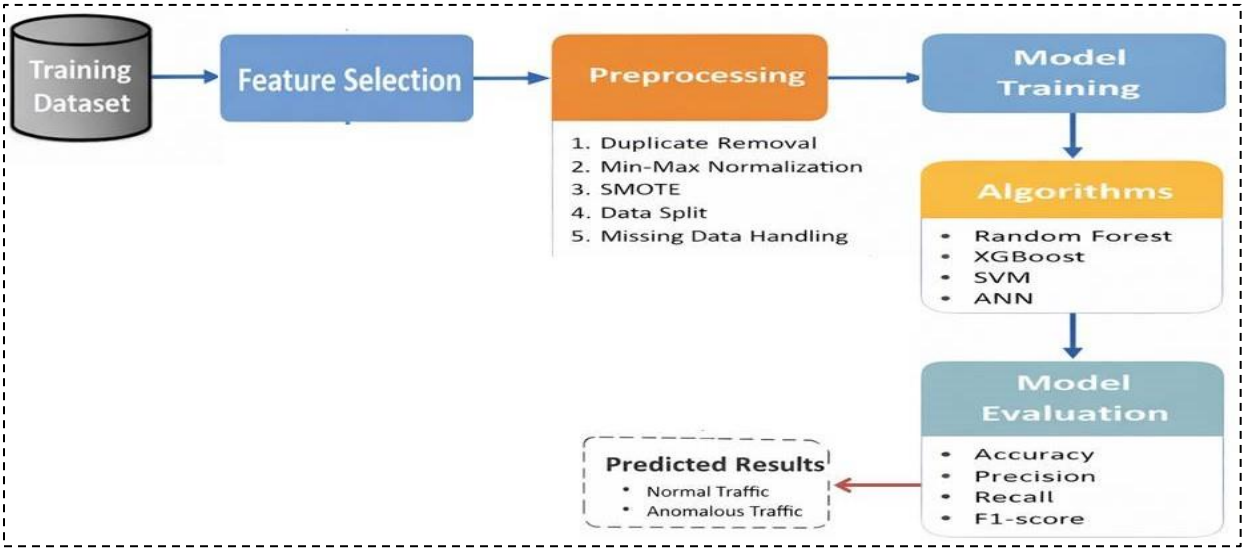

Figure 1: Flow of the Study

3.2 Data Preprocessing

To develop an AI-based model, several preprocessing techniques are needed to clean and prepare the training data. These techniques improve the quality of training data, enabling AI models to concentrate on the patterns which are most important in decision making. Several techniques were used in this work to improve the quality of training data, including handling of missing attributes, data balancing employing SMOTE based strategy, encoding, scaling numerical values of various features, and dataset portioning.

Feature Selection: It is the part of feature engineering and it can be done manually taking help from area experts or built-in scripts can be utilized to show the relationship between various attributes and most interlink attributes can be chosen as features for the said task. Feature selection refers to choosing the most important characteristics—the main elements that affect the goal anomaly—and removing those that have little to no bearing on the decision-making process. In the suggested study, only the Flow_ID and Source_IP were removed off the attribute list.

Incomplete Data Handling: Data is an important factor in ML approaches. Missing data is a

common difficulty in interpreting the training dataset and can be make happen by a variety of factors, such as errors in data gathering, surveys based on incomplete procedures, or anomalies in the original gathered dataset [18]. In the current work, the training set was analyzed to correct for all instances of missing data in order to improve model results.

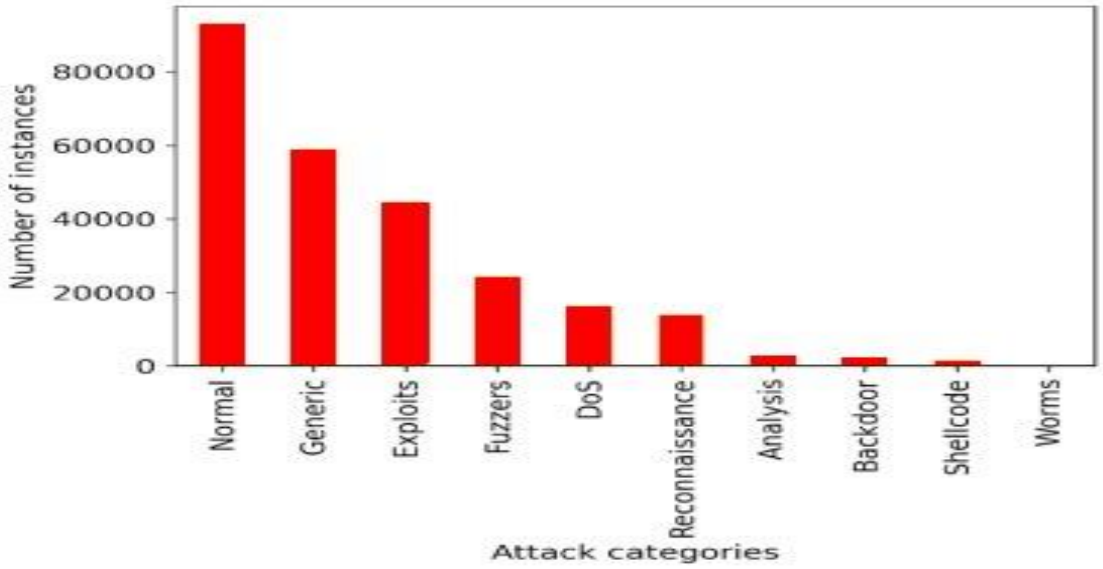

Figure 2: demonstrates the class distribution of benign traffic and various attack types in the dataset.

Features Standardization: This procedure is used to speed up the training process and it guarantees speedy approach towards the minimal loss value. Rescaling in machine learning is a processing of data approach that converts non-normally distributed (SD = 1, average = 0) continuous data points into a distribution that is evenly distributed. Scaling is crucial because in most methods, the degree of feature scaling affects the entire process's effectiveness and convergence duration.

$$Z = \frac{x - u}{\sigma} \qquad (1)$$

Where:

- original value of the feature
- μ = mean (average) of the feature values
- σ = standard deviation of the feature values
- z = standardized value

Feature Encoding: Labelling or feature encoding in machine learning is a process for converting textual attributes into a mathematical representation, enabling algorithms to efficiently handle such feature. Encoding is essential because it enables algorithms to understand categorical input, which enhances their capacity to identify the valuable information needed for decision making and generate highly accurate results. The subject matter of the assignment must be taken into consideration while selecting one of the various ways, each of which has pros and cons of its own. In this study, Label Encoding was used.

Data Splitting: Data partition is the technique of dividing the training dataset into two groups, training and validation. In the proposed study, the training data is split in the ratio of 0.80 and 0.20, respectively, for training and validation.

Balancing Training Data: It is possible to have high accuracy using a machine learning model trained on imbalanced data but with poor performance on key measures like precision and recall. Failure to deal with class imbalance well may result in false predictions and invalid conclusions. Thus, an urgent need to address this issue is necessary at the initial stage in order to

build a robust model which can be generalized. As a means of providing balanced representation within training classes, SMOTE was used. SMOTE also solves the issue of class imbalance by using the interpolation between the existing minority samples to create synthetic ones belonging to the minority class, which allows the model to better learn using under-represented data and minimizes bias on the majority data.

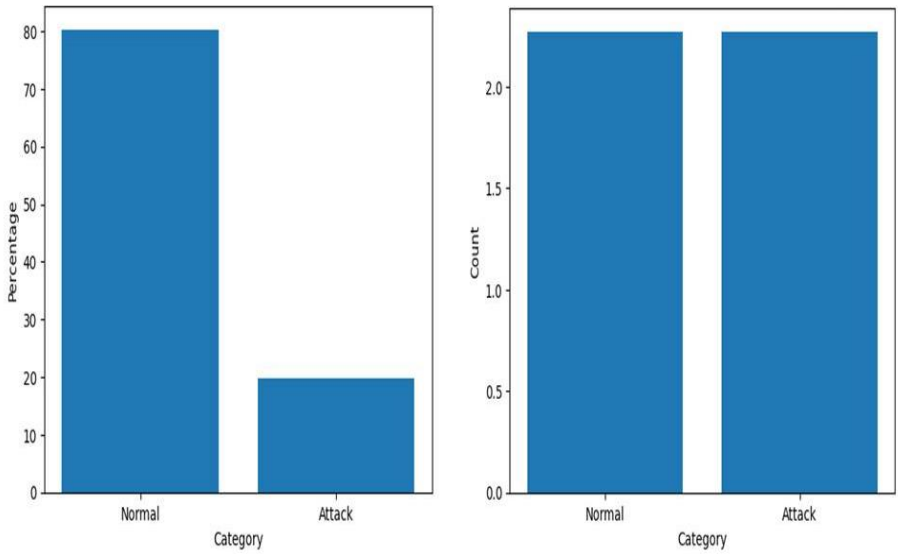

Figure 3: (a) Original Dataset (b) Augmented Dataset

3.3 Machine learning (ML)

ML is regarded as a significant breakthrough in the evolution of AI, serving as the foundation for advanced techniques such as deep neural networks. As a subfield of AI, ML models have the capability to learn from input data and generate predictions, classifications, or predictions for tasks at hand without plain programming. By identifying patterns and relationships within datasets, ML models continuously refine their performance, adapting to new and previously unseen information. This adaptive learning capability enhances accuracy and efficiency, making ML a useful technique for tackling complex tasks across multiple areas, including malicious attacks control and predictive analytics.

Logistic Regression (LR): Logistic Regression (LR) is a statistical method that is popular in binary classification problems, which have two possible outputs (typically 0 or 1). It uses a logistic (sigmoid) curve to graph the correlation between independent variables and the likelihood of a certain outcome, giving outputs that fall in a range of 0 to 1. Logistic regression can only be applied to categorical dependent variables, unlike linear regression, which is adapted to continuous dependent variables. It has been used to perform tasks like customer segmentation and disease prediction in numerous fields like healthcare, finance, and social sciences due to its simplicity and interpretability. Although it is simple, LR still is a strong base model and can be frequently compared with more sophisticated algorithms.

Decision Tree (DT): DT is a directed learning method used in text analysis, based on statistics, and mathematical assumptions. This approach use a categorization or predictive decision tree as a model for prediction to get understandings about a set of text data.

K-Nearest Neighbors: KNNs are types of supervised method of learning utilized in ML. KNN is an un-parametric technique, which means that it makes no assumptions about the input data's true composition. By comparing an entirely new input data point to the K nearest values of the

data in the very first set of data, the KNN approach seeks to classify it. The amount of closest neighbors used in classification, K, is specified by the person who wrote it.

Random Forest (RF): Techniques based on ensemble learning (EL), such as Random Forest (RF), are effective for classification and regression tasks. These methods function by training multiple decision trees (DTs) collectively, enhancing predictive accuracy and robustness. In classification problems, RF determines the final output based on the majority vote among individual trees. EL, as a broader strategy, integrates diverse models to tackle complex problems and optimize network performance. RF algorithms aggregate multiple decision trees, training them on different subsets of input data and averaging their predictions to improve accuracy. The approach prioritizes the most relevant predictions to enhance forecasting precision. When applied to CVD prediction using machine learning (ML), RF demonstrated superior identification performance.

Support Vector Machine (SVM): SVM is an ML technique that employees labelled inputs to classify new data based on training samples [19]. Selection fields, or hyper planes, are utilized to show decision boundaries. To separate a group of data numbers into multiple classes, a hyperspace is utilized. A Radial Basis Function (RBF) removal with a number of one was utilized. SVM tries to classify inputs by creating a mapping that gives every point of a number to the appropriate class or labels, using the reduced number of bias and the largest suitable limit [20].

XGBoost: XGBoost is a widely recognized and freely available machine learning framework used for tasks such as regression, ranking and classification. It is built to be highly effective, robust and scalable while maintaining a lightweight structure. XGBoost generates predictions by combining multiple decision trees through an ensemble learning approach. During the training process, a sequence of trees is constructed, where each new tree focuses on correcting the errors produced by the preceding ones. This iterative boosting strategy ensures XGBoost to efficiently grab intricate and non-linear relationships within the attribute set received from training data, often resulting in higher predictive accuracy compared with many traditional machine learning methods. Due to its strong performance and efficiency, XGBoost is widely adopted in industry and has demonstrated effectiveness across numerous data science applications

In this research, most of the ML techniques were trained and evaluating using their default parameters. Though, for three ML approaches, XGBoost, RF, and SVM, the hyper parameters were tuned to improve their performance, as presented in Table 1.

Table 1: Tuned hyper Parameter Settings of the Machine Learning Models

Algorithm	Parameter	Value
RF	Max_depth	20
	N_estimators	200
	Min_samples_leaf	2
	Min_samples_split	5
SVM	C	1
	Kernel	RBF
	Gamma	0
XGBoost	Sub_sample	0.8
	L_rate	0.05
	Max_depth	6
	N_estimators	200

3.4 Deep Learning (DL)

Deep learning is an advanced version of ML that uses neural networks with multiple layers to handle complex data in a manner that simulates the way the human brain does work. Deep learning networks are effective tools for dealing with big training data and can acquire complex patterns by themselves, without the need for manual feature generations. On the other hand, traditional ML methods are better suited to smaller datasets since they rely on manual attribute nominations and basic computations that can often perform well with less data. While DL can improve precision as well as efficiency in real-world scenarios, it also demands a large training data and computational resources as compared to ML. An ANN is a DL approach designed to grab and recognize meaningful features and patterns from the input data and use them for future decisions. It contained interconnected nodes, or neurons, managed in an approach called layers. ANNs get knowledge from training data through training and adjust weights to improve accuracy and are widely utilized in tasks like image classification, speech processing, and NLP-based jobs. ANNs are deep learning models considered more generalized than ML models and show reliable performance in real-world scenarios. In the proposed study, a five-layer ANN with one input, three hidden, and one final layer was used, surpassing all the experimented models in this study detailed in Table 2.

Table 2: Structural Info of the Proposed Custom ANN Model

Layers	Neuron/ Value	Parameters	Value
Dense_01	264	Learning Rate (Lr)	0.001
Batch Normalization		B_Size	16
Drop Out	0.2	Epochs	100
Dense_02	128	E_Stop	45
Batch Normalization		Activation	Relu
Drop Out	0.2	Optimizer	Adam
Dense_03	64	Loss	Binary_cross_entrophy
Batch Normalization		Train_Set	0.80
Drop Out	0.2	Valid_Set	0.20
Dense (Sigmoid)	1		

4. EXPERIMENTS AND RESULTS

4.1 Training Environment

A training environment provides the necessary computational infrastructure that enables models to learn from data through the use of algorithms, preprocessing techniques, and evaluation tools. Effective research and experimentation require an integrated combination of appropriate software frameworks and hardware resources to ensure efficient model development, training, and performance assessment. The hardware tools used in this work include a Core i7 notebook with sixteen gigabytes of RAM, a 500-gigabyte SSD, and a multi-core processing unit. To build, train, and evaluate the models, a Python tool having multiple libraries like TensorFlow, Keras, Matplotlib, Seaborn, and Scikit-learn were utilized and Jupyter Notebook was used as the coding IDE to run Python programs.

4.2 Evaluation Metrics

Evaluation metrics assess a model's reliability, efficacy, and predictive performance in research. Common measures that assess how successfully a model generalizes to new unseen inputs include precision, recall, accuracy, and F1 score, all of which are employed in the present research. To validate a model, it must be tested using these standardized criteria using data that has never been seen before. These metrics' mathematical representations are given below.

$$\text{Accuracy} = (TP+TN)/(TP+FP+TN+FN) \quad (2)$$

$$\text{Precision} = TP/(TP+FP) \quad (3)$$

$$\text{Recall} = TP/(TP+FN) \quad (4)$$

$$F_1\text{-Score} = 2 \times (\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall}) \quad (5)$$

Here, True positive is denoted by TP, false positive by FP, true negative by TN, and false negative by FN.

4.3 Results and Experiments

The research aims to create a lightweight DL method for timely and accurate network attack prediction so that the anomaly can be handled before it has serious repercussions. In addition to validation results, the study employed both ML and DL techniques, which are detailed below.

4.3.1 Machine Learning (ML)

To design an efficient technique, a large number of experiments are required. Various methodologies were utilized in the current study to design a robust technique that ranged from ML to DL techniques. This section contains machine learning techniques and their results using test data. Five ML models, like DT, XGBoost, KNN, LR, SVM and RF were developed and examined on the proposed test data. All of them scored well with high precision, but RF outperformed all of them with a 99.05% correctness score. Table 3 shows the detailed results of the machine learning (ML) models on the evaluation dataset. Figure 4 shows the confusion matrices of the top three ML models.

Table 3: Evaluation Results of ML Models on Test Data

S No.	Model	Accuracy	Precision	Recall	F_1-Score
1	DT	96.95%	91.66%	93.05%	92.35%
2	KNN	97.40%	92.36%	94.24%	93.29%
3	LR	98.10%	94.44%	96.00%	95.21%
4	SVM	98.55%	95.59%	97.12%	96.35%
5	XGBoost	98.75%	96.03%	97.70%	96.86%
6	**RF**	**99.05%**	**96.82%**	**98.40%**	**97.60%**

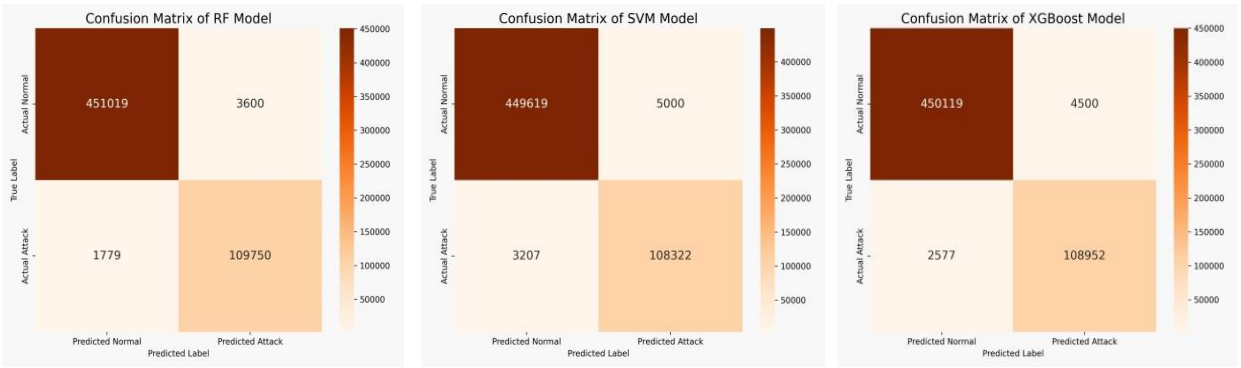

Figure 4: Confusion Matrices of (a) RF (b) SVM (C) XGBoost

4.3.2 Deep Learning (DL)

The proposed DL approach, with an initial layer, three internal layers consisting of 264, 128, and 64 fully connected links respectively, and a final layer for output, was finalized with a test accuracy of 99.05%. Various DL architectures were used in this study, and we altered the layered structure, techniques used for regularization, and width of the ANN model several times before finalization the proposed structure. The test results of the suggested DL model are shown in Table 5. Figure 5 shows the training progress of model training while Figure 6 shows the confusion matrix of the proposed ANN.

Table 5: Results of the Proposed ANN on Validation Set

S No.	Metric	Value (%)
1	Accuracy	99.80
2	Precision	99.21
3	Recall	99.78
4	F_1-Score	99.50

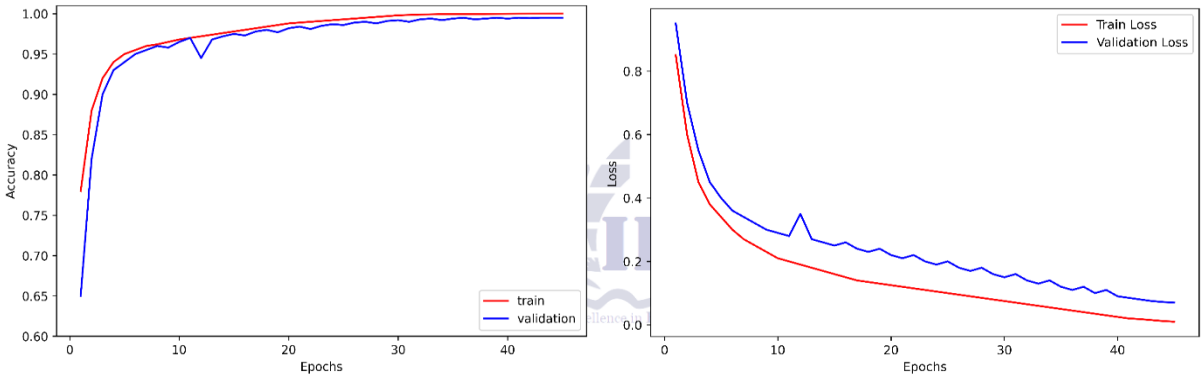

Figure 5: Training Progress of the Proposed DL Model

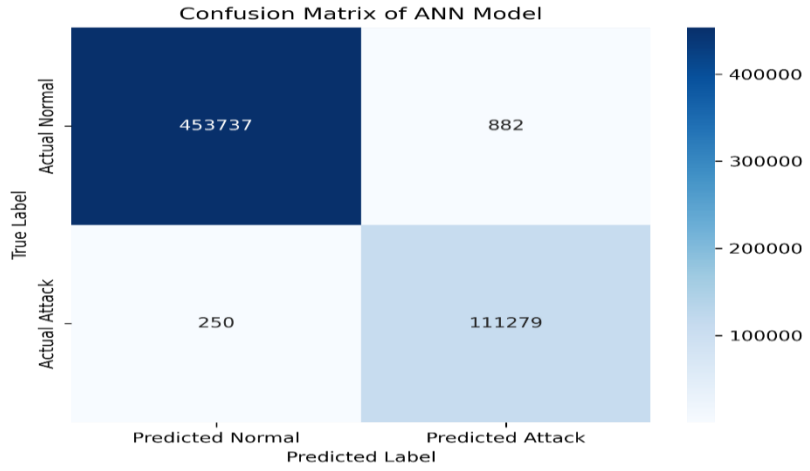

Figure 6: Confusion Matrix of Proposed DL Model

4.4 Comparison of the Proposed ANN with Previous SOTA Techniques

In order to verify the suggested strategy's robustness and effectiveness, it was compared to SOTA approaches in the most recent preceding literature. Table 6 compares the suggested model's findings to earlier methodologies.

Table 6: Comparison of the proposed ANN with Previous Methods

S. No.	Author	Dataset Balancing	Model	Accuracy (%)
1	Moualla et al. [9]	SMOTE	ELM	98.43
2	Sun & Kasongo [21]	–	DT	90.85
		–	ANN	77.51
3	Ahmad et al. [12]	–	RF	98.67
4	Kasongo & Sun [22]	–	FFDNN	87.10
5	Choudhary & Kesswani [23]	–	DNN	91.50
6	Aleesa et al. [24]	–	ANN	97.89
7	Dener & AI [25]	STL	RNN+CNN	98.03
8	Zhang et al. [26]	SGM	CNN	96.54
9	Hassan et al. [27]	–	CNN-LSTM	97.17
10	**Proposed**	**SMOTE**	**RF**	**99.05**
		SMOTE	**ANN**	**99.80**

4.5 Discussion

In the proposed study, several ML technique including SVM, RF, XGBoost, LR, DT, and KNN, were trained to identify network attacks. Among these models, the RF model demonstrated the best performance, achieving an accuracy of 99.05%, outperforming the other ML approaches. The RF model was trained with hyper parameter optimization, as presented in Table 2. Prior to model building and fine-tuning, the training set was balanced utilizing the SMOTE technique to mitigate bias toward the larger category, since the initial dataset is imbalanced as shown in Figure 2. The primary objective of this study was to explore and evaluate DL–based models, which are generally recognized for providing more accurate and generalized outcomes in contrast to traditional ML techniques; therefore, the ML techniques were utilized as baseline approaches.

The proposed research evaluated several DL architectures, including Recurrent Neural Networks (RNNs), one-dimensional CNN (1D-CNNs), ANNs, and LSTM networks. Compared with ML approaches, DL techniques are often considered data-intensive, requiring larger datasets due to their complex architectures. In this study, the training dataset was carefully utilized during the training of DL models. In comparison to advanced DL models, ANN performed well, needing less training time and computational resources. Several iterative modifications were made to the ANN architecture, including adjustments to layer configurations and fundamental parameters; however, many of these configurations did not yield satisfactory results for the problem at hand. As noted earlier, such complex frameworks generally require larger volumes of training data to perform optimally. Consequently, a computationally efficient ANN architecture was adopted, consisting of an initial layer, three internal layers, and an output layer. The suggested ANN approach surpassed all machine learning techniques, obtaining an accuracy of 99.80%. Figure 6 shows the graphical comparison of the aforementioned models.

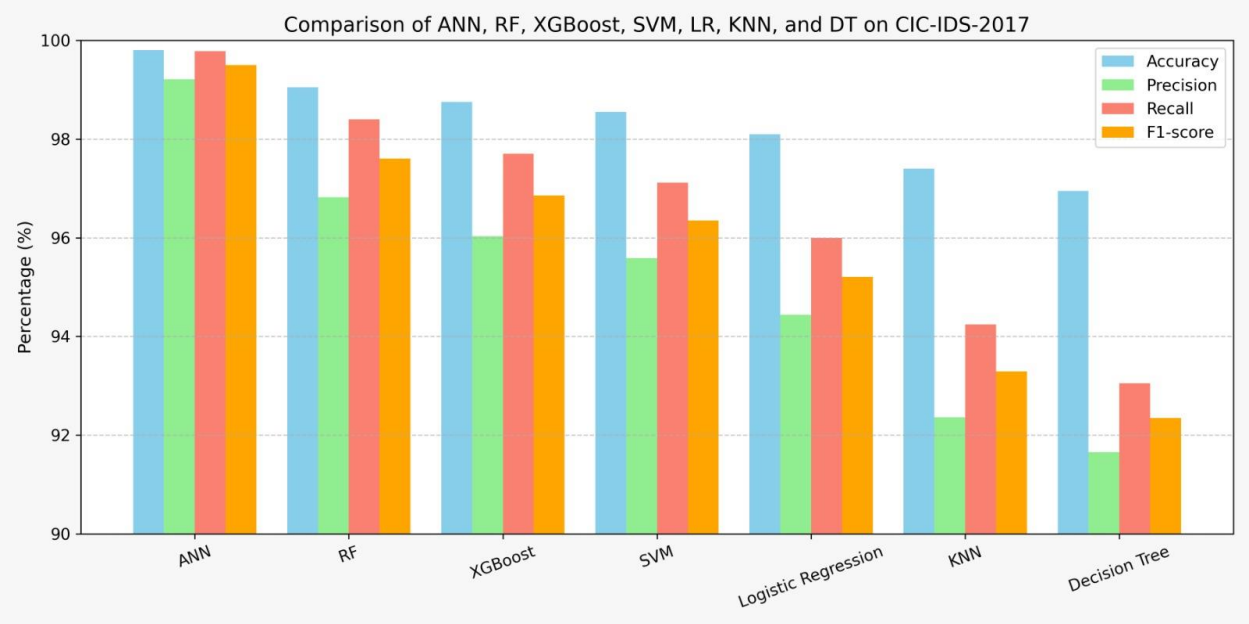

Figure 6: Shows the Graphical Comparison of the Aforementioned Models

5. CONCLUSION

This study presented a lightweight and efficient intrusion detection framework for SDN using both ML and DL techniques. Evaluation outcomes showed that the proposed ANN outperformed all baseline models, achieving a superior accuracy of 99.80%, while RF showed the best performance among ML models with 99.05% accuracy. The integration of preprocessing techniques, particularly SMOTE, significantly improved model generalization by addressing class imbalance. The findings confirm that deep learning models, even with relatively simple architectures, can effectively capture complex network traffic patterns. Moreover, the proposed ANN offers a favorable trade-off between computational efficiency and predictive performance and highlights the potential of lightweight DL-based approaches for robust and real-time intrusion detection in SDN environments.

Future work will focus on incorporating more realistic data balancing strategies by utilizing multiple publicly available datasets to enhance model generalization across diverse network environments. Additionally, exploring novel DL architectures, such as hybrid and attention-based techniques, may further enhance detection performance. The integration of real-time data patterns and online learning mechanisms can also be considered to adapt to evolving cyber threats. These directions will contribute to developing more robust and scalable intrusion detection systems.

REFERENCES

[1] Hakiri, A., Gokhale, A., Berthou, P., Schmidt, D. C., & Gayraud, T. (2014). Software-defined networking: Challenges and research opportunities for future internet. Computer Networks, 75, 453-471.

[2] Bojović, Ž. C., & Bojović, P. D. (2026). Software-defined networking: architecture, protocols, and applications. Foundations and Trends® in Networking, 15(1-3), 1-271.

[3] Jammal, M., Singh, T., Shami, A., Asal, R., & Li, Y. (2014). Software defined networking: State of the art and research challenges. Computer Networks, 72, 74-98.

[4] Hossain, M. A., & Islam, M. S. (2023). Ensuring network security with a robust intrusion detection system using ensemble-based machine learning. Array, 19, 100306.

[5] Diana, L., Dini, P., & Paolini, D. (2025). Overview on intrusion detection systems for computers networking security. Computers, 14(3), 87.

[6] Hamidou, S. T., & Mehdi, A. (2025). Enhancing IDS performance through a comparative analysis of Random Forest, XGBoost, and Deep Neural Networks. Machine Learning with Applications, 100738.

[7] Koukaras, P., & Tjortjis, C. (2025). Data Preprocessing and Feature Engineering for Data Mining: Techniques, Tools, and Best Practices. AI, 6(10).

[8] Elezmazy, I., & Abdullah, W. (2024). Advanced Intrusion Detection in Software-Defined Networks through Ensemble Modeling. Information Sciences with Applications, 4, 1-11.

[9]Moualla, S., Khorzom, K., & Jafar, A. (2021). Improving the performance of machine learning-based network intrusion detection systems on the UNSW-NB15 dataset. *Computational Intelligence and Neuroscience, 2021*, 1–13.

[10] Nimbalkar, P., & Kshirsagar, D. (2021). Feature selection for intrusion detection system in Internet-of-Things (IoT). *ICT Express, 7*(2), 177–181.

[11] Kumar, V., Das, A. K., & Sinha, D. (2020). Statistical analysis of the UNSW-NB15 dataset for intrusion detection. In *Computational intelligence in pattern recognition* (pp. 279–294). Springer.

[12] Ahmad, M., Riaz, Q., Zeeshan, M., et al. (2021). 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, 2021*(1), 1–23.

[13] Mugabo, E., Zhang, Q. Y., Ngaboyindekwe, A., et al. (2021). Intrusion detection method based on MapReduce for evolutionary feature selection in mobile cloud computing. *International Journal of Network Security, 23*(1), 106–115.

[14] Talita, A., Nataza, O., & Rustam, Z. (2021). Naïve Bayes classifier and particle swarm optimization feature selection method for classifying intrusion detection system dataset. *Journal of Physics: Conference Series, 1812*, 012021.

[15] Seth, S., Singh, G., & Kaur Chahal, K. (2021). A novel time-efficient learning-based approach for smart intrusion detection system. *Journal of Big Data, 8*(1), 1–28.

[16] Stiawan, D., Idris, M. Y. B., Bamhdi, A. M., et al. (2020). CICIDS-2017 dataset feature analysis with information gain for anomaly detection. *IEEE Access, 8*, 132911–132921.

[17] Sweety. (2017). CICIDS2017 Full dataset [Data set]. Kaggle. https://www.kaggle.com/datasets/sweety18/cicids2017-full-dataset?select=combine.csv

[18] Y. Muhammad, M. Tahir, M. Hayat, and K. T. Chong, "Early and accurate detection and diagnosis of heart disease using intelligent computational model," Sci. Rep., vol. 10, no. 1, pp. 1–18, 2020, doi: 10.1038/s41598-020-76635-9

[19] Jeena RS, Kumar S. Stroke prediction using SVM, International Conference on Control. Instrumentation, Communication and Computational Technologies (ICCICCT), 2016: 600–602

[20] Rahman MM, Rana MR, Alam NAA, Khan MSI. A web-based heart disease prediction system using machine learning algorithms; 2022 June; 12. 64-80.

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

[22] Kasongo, S. M., & Sun, Y. (2020). A deep learning method with wrapper-based feature extraction for wireless intrusion detection systems. *Computers & Security*, 92, 101752.

[23] Choudhary, S., & Kesswani, N. (2020). Analysis of KDD Cup'99, NSL-KDD, and UNSW-NB15 datasets using deep learning in IoT. *Procedia Computer Science*, 167, 1561–1573.

[24] Aleesa, A., Younis, M., Mohammed, A. A., et al. (2021). Deep intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques. *Journal of Engineering Science and Technology*, 16(1), 711–727.

[25] Al, S., & Dener, M. (2021). STL-HDL: A new hybrid network intrusion detection system for imbalanced datasets in big data environments. *Computers & Security*, 110, 102435.

[26] Zhang, H., Huang, L., Wu, C. Q., et al. (2020). An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced datasets. *Computer Networks*, 177, 107315.

[27] Hassan, M. M., Gumaei, A., Alsanad, A., et al. (2020). A hybrid deep learning model for efficient intrusion detection in big data environments. *Information Sciences*, 513, 386–396.