

# Deep learning-based evaluation for distributed denial of service attacks detection

Neethu S.<sup>1</sup>, H. V. Ravish Aradhya<sup>2</sup>, Viswavardhan Reddy Karna<sup>3</sup>

<sup>1</sup>Department of Computer Science Engineering, RV College of Engineering, Visvesvaraya Technological University, Belagavi, India

<sup>2</sup>Department of Electronics and Communication Engineering, RV College of Engineering, Visvesvaraya Technological University, Belagavi, India

<sup>3</sup>Department of Artificial Intelligence and Machine Learning, RV College of Engineering, Visvesvaraya Technological University, Belagavi, India

## Article Info

### Article history:

Received Jun 21, 2024

Revised Aug 4, 2025

Accepted Sep 7, 2025

### Keywords:

Deep learning

Distributed denial of service attack

Openflow protocol

SMOTE

Software-defined networks

## ABSTRACT

Software-defined network (SDN) introduces a programmable and centralized control mechanism for managing network infrastructure, enhancing flexibility and efficiency. However, this architecture is prone to security threats, particularly distributed denial of service (DDoS) attacks that exploit centralized control. This study presents a comparative analysis of several deep learning (DL) models—namely, multilayer perceptron (MLP), artificial neural network (ANN), convolutional neural network (CNN), recurrent neural network (RNN), and long short-term memory (LSTM)—for detecting DDoS threats within SDN environments. The research incorporates key preprocessing techniques such as feature selection and synthetic minority oversampling technique (SMOTE) to handle class imbalance. The results indicate that sequence-aware models like LSTM and RNN are highly effective in interpreting temporal network behavior, with LSTM achieving the highest performance (accuracy: 91%, precision: 86%, recall: 94%, and F1-score: 90%). These findings underscore the potential of advanced DL methods in fortifying SDN infrastructures against complex cyber threats.

This is an open access article under the [CC BY-SA](#) license.



## Corresponding Author:

Neethu S.

Department of Computer Science Engineering, RV College of Engineering

Visvesvaraya Technological University

Belagavi, Karnataka, India

Email: neethus@rvce.edu.in

## 1. INTRODUCTION

Software-defined network (SDN) has significantly transformed the landscape of network management by decoupling the control and data planes, enabling centralized and programmable control over network infrastructure. This architectural advancement fosters dynamic and efficient communication but also introduces novel security vulnerabilities. In particular, the centralized control mechanism becomes a critical point of failure, susceptible to targeted attacks such as botnets and distributed denial of service (DDoS) threats, which can compromise both control and data plane components. DDoS attacks remain a persistent and escalating challenge in SDN environments due to the intricate and dynamic nature of network traffic. By flooding the target system with fake requests, attackers aim to exhaust computational resources and disrupt legitimate services. These attacks are not only stealthy and inexpensive to execute but also capable of inflicting substantial damage on network performance and availability [1], [2]. To address these concerns, researchers have proposed leveraging deep learning (DL) techniques for real-time traffic analysis and

anomaly detection. One such approach involves the use of deep neural networks (DNNs), which offer scalable architectures for identifying complex traffic patterns associated with DDoS activities [3]. Several hybrid models have also been developed—for instance, a combination of convolutional neural networks (CNNs) and bidirectional long short-term memory (BiLSTM) networks—which enhance detection capabilities using well-established datasets such as UNSW-NB15 and NSL-KDD for both binary and multiclass classification tasks [4]. Innovations in anomaly detection continue to emerge. Feed forward convolutional neural networks (FFCNNs) have been applied to low-rate denial of service (DoS) scenarios in IoT-SDN settings, often paired with feature selection techniques like support vector machine (SVM)-based wrappers. These models are benchmarked against classical classifiers, including J48, random forests, reduced error pruning (REP) tree, and multi-layer perceptron (MLP) [5]. Cloud-integrated DL architectures have also been proposed for detecting and mitigating phishing and botnet attacks at scale [6]. Moreover, strategies such as moving target defense (MTD) have been introduced to divert malicious traffic toward decoy systems, effectively reducing the impact on primary servers [7]. A vast body of literature explores various machine learning (ML) and DL-based solutions to detect DDoS threats within SDN infrastructures [8]–[12]. For instance, a two-tiered approach employing entropy-based anomaly detection followed by CNN-based packet classification has shown promise in differentiating legitimate and suspicious flows [13]. Other contributions include sparse autoencoders combined with DNNs for feature representation and classification [14], as well as integrated intrusion detection systems (IDS) and deep reinforcement learning (DRL)-based intrusion prevention systems (IPS) tailored for low-rate DDoS attacks [15]. To strengthen SCADA systems built on SDN, advanced architectures utilizing recurrent neural networks (RNNs), long short-term memory (LSTM), and gated recurrent units (GRUs) have also been proposed, enhancing detection accuracy across various deployment scenarios [16]–[18].

A method leveraging spatial-temporal graph convolutional networks (ST-GCN) has been introduced for securing the data plane of SDNs. This model utilizes in-band network telemetry (INT) with sampling to monitor network state and pinpoint the switches involved in forwarding DDoS traffic flows [19]. In another approach focused on the early detection of TCP SYN flood attacks, an extended chi-square goodness-of-fit test is employed. This technique evaluates the distribution of half-open connections by calculating the p-value, which helps detect anomalies in network behavior [20]. Further, authors proposed a DNN-based model offering a scalable and effective framework for detecting DDoS attacks in SDN environments, demonstrating superior accuracy and reliability across diverse datasets and real-world conditions [21]. Additionally, LSTM and hybrid CNN-LSTM architectures have been employed to design IDS, particularly using the CIC-DDoS2019 dataset for training and validation [22]. A novel ensemble method called SE-IDS combines decision boundaries from five tree-based classifiers, with a MLP serving as the meta-learner for final classification [23]. Another notable development includes the cascade forward back propagation neural network (CFBPNN), which utilizes a refined subset of features selected using correlation-based feature selection (CFS). This model has been validated across multiple datasets [24]. Moreover, anomaly-based IDS for IoT-enabled SDN environments have also been proposed, utilizing CNN models to examine traffic patterns and detect suspicious behavior effectively [25].

This study aims to comparatively assess a variety of DL models for their ability to detect and mitigate DDoS attacks in SDN settings. A critical component of this research involves optimizing feature selection from network traffic data, which can significantly enhance the accuracy and efficiency of these models while minimizing computational load on the SDN controller. Summary of key contributions:

- i) Exploratory data analysis (EDA): a thorough analysis was conducted to examine all dataset features, including their types, ranges, distributions, and any missing or anomalous values.
- ii) Data cleaning: missing values were handled using techniques such as imputation or row exclusion, depending on their impact on the analysis.
- iii) Descriptive statistics: measures such as mean, median, standard deviation, and interquartile range were calculated to understand distribution patterns. For instance, a significant gap between mean and median packet counts could indicate skewness due to outliers.
- iv) Data visualization:
  - Boxplots revealed outliers that could signify attack instances.
  - Scatter plots highlighted relationships between continuous variables.
  - Bar charts displayed categorical distributions, such as protocol types and port usage.
- v) Addressing class imbalance: the synthetic minority over-sampling technique (SMOTE) technique was applied to synthetically balance underrepresented classes, thereby reducing model bias toward majority class instances.
- vi) Feature selection and importance: correlation matrices and the random forest algorithm were employed to assess and select the most influential features for training DL models.
- vii) DL models used: various DL models were tested, including MLP, artificial neural network (ANN), CNN, RNN, and LSTM, each chosen for their ability to model complex patterns in network traffic.

- viii) Model optimization and validation: each model underwent hyperparameter tuning and cross-validation to ensure generalization and prevent overfitting.
- ix) Performance evaluation: the models were evaluated using accuracy, precision, recall, and F1-score, offering insights into detection efficacy and computational efficiency.
- x) DDoS mitigation via graph theory: the SDN controller periodically collects flow statistics (e.g., every 5 seconds) from OpenFlow switches. These are analyzed using a trained DL model to identify suspicious flows. When attacks are detected, the controller triggers mitigation steps based on graph theory to minimize disruption while maintaining service quality.

The remaining sections of the paper cover the following topics: section 2 details the datasets used in experimentation, the architecture and methodology of the proposed detection system. Section 3 discusses results, metrics, and comparisons with contemporary solutions. Finally, section 4 concludes the paper and suggests directions for future research.

## 2. METHOD

### 2.1. Experimental setup

The experimental framework was constructed using the Mininet network emulator in conjunction with the RYU SDN controller, as shown in Figure 1. The simulations were executed using Mininet version 2.3.2, which provides robust support for Open vSwitch (OVS), a widely adopted virtual switch compatible with OpenFlow protocols. The experiments were run on a system equipped with an Intel Core i7-5500U processor, 8 GB of RAM, and Windows 10 as the operating system. DL models were developed using the Keras library in Python. A customized SDN topology was implemented, featuring a hierarchical tree-based structure managed by a centralized controller. This topology integrated seven OpenFlow switches, with one switch (S1) connected to six hosts, and the remaining switches connected to three hosts each. The network architecture consisted of three layers: the control plane (housing the SDN controller), the data plane (containing hosts and switches), and the application layer, which included four components: flow statistics collection, feature extraction, a DL classifier, and an attack mitigation module. After deploying the topology, connectivity was validated using the ping command across all hosts. Traffic flows were generated using TCP with packet sizes fixed at 512 bytes and flow durations set to 12,000 milliseconds. The packet count for flows ranged between 500 and 1000, forming the basis for the dataset used in training and evaluation.

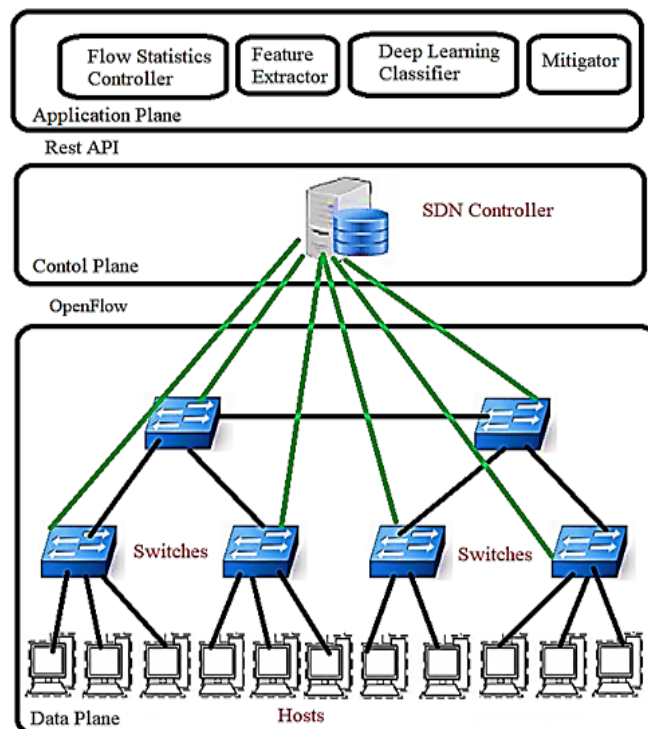


Figure 1. A schematic diagram of SDN

The dataset employed in this study comprises 104,346 entries distributed across 23 columns, encompassing both features and target labels. These columns fall into the following categories:

- Feature attributes: these represent various network traffic indicators used to differentiate between benign and malicious (DDoS) activities. Notable features include packet size, transmission rate, byte count, source and destination IP addresses, and several other statistical measures relevant to traffic flow analysis.
- Label column: this serves as the classification target, identifying whether each data instance corresponds to legitimate traffic or a DDoS attack. This label is critical for supervised ML, allowing the models to learn from historical examples.
- Data partitioning: to facilitate DL, the dataset is split into three subsets. Seventy-five percent (75%) of the records are allocated for training, while the remaining 25% are divided between validation and testing to assess model generalizability and performance.

It is worth noting that many of the traffic-related features, such as packet size, rate, and source/destination addresses, are instrumental in capturing the unique patterns associated with DDoS behaviors.

## 2.2. Exploratory data analysis

The preliminary analysis of the dataset involves both statistical summarization and feature examination. The summary statistics for raw data are presented in Table 1, while Table 2 lists the selected features used in model development after feature engineering. The dataset contains 8,714 records, each comprising multiple variables that provide a detailed view of network traffic behavior. Among these, the 'switch' attribute likely denotes identifiers for network switches. Its mean value of 11,089.83 may suggest that the data is either encoded or scaled, as switch identifiers are typically categorical. The variables 'pktcount' and 'bytecount' represent the number of packets and total bytes per event, respectively. On average, each event has 2.32 packets, suggesting consistent packet flow across records. Conversely, the average byte count—approximately 79 million bytes—reflects a broader variation in traffic size. Time-based features such as 'dur', 'dur\_nsec', and 'tot\_dur' provide event durations in seconds, nanoseconds, and as an aggregate measure. These indicate a broad spectrum of traffic durations. The 'flows' variable, averaging 3.32 per event, hints at multiple concurrent communication sessions.

Table 1. Dataset

	pktcount	bytecount	dur	dur_nsec	tot_dur	packetins	pktperflow
count	14.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00
mean	7.47E+04	7.91E+07	194.01	5.48E+08	1.95E+11	1,920.42	1.05E+04
std	4.07E+04	4.31E+07	117.35	2.32E+08	1.17E+11	254.13	4.05E+03
min	284.00	3.03E+05	-	7.90E+07	8.37E+08	558.00	0.00E+00
25%	3.73E+04	3.96E+07	100.00	3.91E+08	1.01E+11	1,931.00	8.64E+03
50%	7.66E+04	8.17E+07	190.00	5.56E+08	1.91E+11	1,943.00	1.34E+04
75%	1.13E+05	1.20E+08	280.00	7.26E+08	2.81E+11	1,943.00	1.35E+04
max	1.35E+05	1.44E+08	473.00	9.14E+08	4.73E+11	2,242.00	1.37E+04

Table 2. Feature selection in a dataset

	byteperflow	pktrate	tx_bytes	rx_bytes	tx_kbps	rx_kbps	tot_kbps
count	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00	8,714.00
mean	1.11E+07	349.86	4.82E+07	4.81E+07	873.43	873.11	1,746.54
std	4.35E+06	134.91	1.51E+08	1.10E+08	2,848.97	2,269.07	3,426.36
min	0.00E+00	-	2.85E+03	9.26E+02	-	-	-
25%	9.21E+06	288.00	3.59E+03	1.47E+03	-	-	-
50%	1.43E+07	446.00	3.84E+03	3.54E+03	-	-	-
75%	1.44E+07	451.00	4.25E+03	6.16E+06	-	-	2.57E+03
max	1.46E+07	456.00	1.27E+09	9.91E+08	2.06E+04	1.66E+04	2.06E+04

The 'packetins' feature, possibly measuring incoming packets, shows an average of 1,920.42, suggesting significant variability in packet reception rates. Metrics like 'pktperflow' and 'byteperflow' capture per-flow transmission statistics, revealing diverse patterns through their high standard deviations, critical for identifying anomalous behaviors. Additionally, 'pktrate' exhibits a mean value of 349.86, with wide variation, potentially reflecting both idle and active communication periods in the traffic logs. To enhance model performance, feature engineering was employed—an essential step involving domain-driven transformations to highlight meaningful patterns in the data. Linear associations between features were uncovered through correlation analysis, while random forest was used to compute feature importance, facilitating dimensionality reduction and directing model attention to the most informative variables.

Given the typical imbalance in DDoS-related datasets, the SMOTE was applied. This method generates synthetic samples of underrepresented attack traffic, thus ensuring that the classifier does not become biased toward the dominant (benign) class. A diverse set of DL models was then chosen for evaluation, ranging from fully connected networks to architectures tailored for sequential data. This variety was key to capturing different aspects of the traffic patterns. Overall, the entire pipeline—from exploratory analysis and preprocessing to model selection—was designed to ensure a robust, comprehensive approach to DDoS attack detection, with an emphasis on reliability and predictive effectiveness.

### 2.3. Model development

A range of well-established DL models is utilized in this study, each trained on the dataset generated from SDN-based simulations. These models are particularly effective in identifying intricate and non-linear data patterns—capabilities that often surpass those of conventional ML algorithms. The overall workflow and methodology for detecting and mitigating DDoS attacks using these DL models are illustrated in Figure 2. Various DL models used for classification are described in the following sub-section.



Figure 2. Schematic diagram of DDoS attack detection and mitigation using DL models

#### 2.3.1. The multi-layer perceptron classifier

MLP classifier is a type of feed-forward ANN composed of several layers of nodes, each applying a nonlinear activation function. The specific MLP classifier in question is set up with two hidden layers containing 100 and 50 nodes, respectively. This design aims to identify complex patterns in the data through multiple levels of abstraction. The parameter `max_iter = 1000` allows the model up to 1000 iterations to converge on a solution, unless it meets a stopping criterion earlier. This extensive number of iterations is advantageous for intricate datasets where the relationships between inputs and outputs are challenging to model. Using a `random_state` ensures the results are reproducible by fixing the seed for the random number generator used in initializing weights. Training the MLP on resampled data, likely adjusted to correct class imbalances through techniques like SMOTE, helps the classifier perform well for both minority and majority classes. The MLP's capability to capture non-linear relationships makes it especially effective for tasks such as predicting DDoS attacks in SDN environments, where attack patterns might be subtle and not linearly separable.

#### 2.3.2. Artificial neural network model

The ANN model, implemented with TensorFlow's Keras, features a sequential architecture with two layers. The first layer is a dense layer with 64 neurons using the rectified linear unit (ReLU) activation function, aiding in non-linearity and mitigating the vanishing gradient problem. The output layer has one neuron with a sigmoid activation function, suitable for binary classification tasks like detecting DDoS attacks. The model employs `binary_crossentropy` as the loss function and the Adam optimizer. An `EarlyStopping` callback with a patience of 10 epochs and `restore_best_weights` option helps prevent overfitting and ensures the model generalizes well. The model is trained on resampled data to address class imbalance.

#### 2.3.3. Convolutional neural network model

CNN is tailored for one-dimensional sequence data, such as time series or network traffic flow analysis. The architecture includes convolutional layers with 32 and 64 filters and a kernel size of 3, which extract high-level features by applying filters across the input data to capture local dependencies within sequences. Each convolutional layer is followed by a MaxPooling layer with a pool size of 2, which downsamples the input representation, reduces dimensionality, and enhances model performance by introducing translational invariance. Following the convolutional and pooling operations, the feature maps are flattened into a one-dimensional array, enabling integration with fully connected (dense) layers. These thick layers perform additional transformations before producing the final classification output. A sigmoid activation function is applied in the output layer to support binary classification tasks, such as distinguishing between regular and DDoS traffic. The model is compiled using the binary cross-entropy loss function, paired with the Adam optimizer to ensure efficient and adaptive gradient updates during training. To

safeguard against overfitting, an early stopping mechanism is implemented, configured with a patience of 10 epochs, allowing training to halt if no performance improvement is observed, thus promoting better generalization to unseen data.

#### 2.3.4. Recurrent neural network model

RNN model with SimpleRNN layers excels at handling sequences where current outputs depend on previous computations, making it suitable for analyzing sequential data like network traffic. The model features a SimpleRNN layer with 50 units, capable of capturing temporal dynamics but potentially struggling with long-term dependencies due to the vanishing gradient problem inherent in basic RNNs. The RNN layer utilizes the ReLU activation function, enabling the network to capture nonlinear relationships and extract complex temporal features from sequential data. For binary classification purposes, the output layer employs a sigmoid activation function, effectively mapping the output to a probability score. The model is compiled using the Adam optimizer, known for its adaptive learning capabilities, along with the binary cross-entropy loss function, which is a standard choice for handling binary classification problems. Early stopping is used during training, monitoring validation accuracy, and halting the process if no improvement occurs over several epochs, preventing overfitting and ensuring the model generalizes well. The RNN architecture is well-suited for real-time streaming data, where recent data points are critical for predictions. However, for very long sequences or dispersed important information, LSTM or GRU models may be more effective due to their advanced gating mechanisms.

#### 2.3.5. Long short-term memory model

The LSTM model, an advanced RNN architecture, excels at learning long-term dependencies, crucial for sequential data with important temporal features. It includes an LSTM layer with 50 units, allowing it to retain information over extended periods, which is essential for network traffic sequences. The input shape matches the reshaped training data, presenting network traffic as a sequence. LSTM avoids the vanishing gradient problem, making it ideal for complex sequences like network traffic data. The output layer comprises a single neuron activated by a sigmoid function, making it well-suited for binary classification by producing a probability score between 0 and 1. The model is compiled using the Adam optimizer, which ensures efficient training through adaptive learning rates, and the binary cross-entropy loss function, a commonly used criterion for evaluating performance in binary classification tasks. Early stopping monitors validation loss, halting training when there's no improvement, and reverting to the best model weights, preventing overfitting and ensuring generalization to unseen data.

### 2.4. Performance metrics

To assess the effectiveness of each DL model, a variety of evaluation metrics suitable for classification tasks were employed. These include accuracy, precision, recall, and F1-score, offering a comprehensive perspective on each model's predictive capabilities.

- Accuracy reflects the overall correctness of the model and is calculated as the proportion of correctly predicted instances to the total number of predictions made.

$$Accuracy = \frac{(TP+TN)}{(TP+FP+TN+FN)} \quad (1)$$

Precision quantifies the ratio of accurate positive detections to the total instances that were predicted as positive. It measures the model's ability to avoid false alarms when identifying attack traffic.

$$Precision = \frac{TP}{(TP+FP)} \quad (2)$$

- Recall (also known as sensitivity or the actual positive rate) gauges how well the model identifies actual attack cases. It is computed by dividing the number of true positives by the sum of true positives and false negatives.

$$Recall = \frac{TP}{(TP+FN)} \quad (3)$$

F1-score provides a balanced measure by computing the harmonic mean of precision and recall. This metric is handy in cases of class imbalance, ensuring both false positives and false negatives are adequately considered.

$$F_1 = 2 \times \frac{(precision \times recall)}{(precision + recall)} \quad (4)$$

## 2.5. DDoS mitigation strategy using graph-based dynamic flow control

In the SDN environment, the controller plays a pivotal role in monitoring ongoing traffic and ensuring protection against malicious intrusions. Upon identifying suspicious behavior, it must swiftly activate a defense mechanism to limit the impact and maintain seamless network performance. While conventional solutions often involve filtering or blocking malicious traffic, they typically leave behind residual flow entries within the switches. These leftover entries can hinder packet forwarding and impose unnecessary processing overhead on both the controller and the switches.

To counter this limitation, we propose a graph-theory-based mitigation mechanism that incorporates dynamic flow deletion. In our approach, the controller periodically (e.g., every 5 seconds) collects flow statistics from associated OpenFlow switches. These statistics, containing vital traffic characteristics, are fed into a pre-trained DL classifier to determine whether the flow is benign or indicative of an attack.

Suppose the classifier flags a flow as malicious. In that case, the controller logs this in a "gray list" ( $S_{\text{gray}}$ ), isolating the suspect flows from those originating in switches identified as carrying attack traffic. This mechanism allows for deeper analysis and reduces the risk of prematurely dropping legitimate packets. The controller continues to re-route flows in the gray list through the classifier for additional verification.

A counter maintains a tally of detected malicious flows, and a predefined threshold helps determine when further action is necessary. Once this threshold is met, the controller creates two additional lists:

- Delete list ( $S_{\text{d}}$ ): contains flow entries scheduled for removal.
- Block list ( $S_{\text{b}}$ ): includes hosts identified as malicious, storing attributes such as MAC/IP addresses, port numbers, and ingress details for future reference.

Using its host tracking capabilities, the controller gathers identifying information about the attacking sources. Upon reaching the attack flow threshold, a graph-theoretic tracing algorithm is invoked to reconstruct the attack path. This involves identifying the sequence of switches (hops) through which the malicious traffic traversed. The following expression represents the attack path:

$$E_{i,j} = \sum(s_i, r_i) \rightarrow (s_j, r_j), \text{ where } s_i, s_j \in S_{\text{attack}} \quad (5)$$

Here,  $E_{(i,j)}$  denotes an edge in the attack graph, and  $S_{\text{attack}}$  is the set of switches involved in routing the DDoS traffic. If traffic flows through both switches  $s_i$  and  $s_j$  with valid forwarding rules, a connection (edge) between them is established. The central objective of this approach is to pinpoint the exact attack path, thereby allowing targeted dropping of malicious traffic. We assume that switches closer to the source of the attack carry a higher concentration of malicious packets. As a result, edge switches (where attack traffic enters) are assigned higher dropping rates, whereas intermediate switches receive lower rates to avoid collateral damage to legitimate traffic. A dropping rate for each switch is computed using traffic-based indicators. If a switch is only handling clean traffic, no dropping is enforced. For switches under suspicion, the drop rate is determined using the following formula:

$$\text{redge} = k(\Delta H, \Delta N) \quad (6)$$

Where  $\Delta H$  is the change in entropy of source IP addresses over time, and  $\Delta N$  is the change in packet count over time at the switch.

Finally, the controller sends an OFPFC\_ADD message to the affected switches, inserting new flow rules that drop traffic as per the delete list ( $S_{\text{d}}$ ) and calculated drop rates. If a host's dropping rate reaches 100%, it is added to the blocklist. All future traffic from that host is blocked, thereby neutralizing the attack at its source.

## 3. RESULTS AND DISCUSSION

The evaluation of various DL models applied to DDoS attack detection within an SDN framework highlights distinct performance trends across different algorithms. A detailed comparison is provided in Table 3, which summarizes the classification effectiveness of each model based on key performance metrics. This comprehensive analysis offers insights into how well each DL approach identifies and mitigates malicious traffic. The MLP model has a low accuracy of 0.48 but an exceptionally high recall of 0.99, indicating it is susceptible and correctly identifies nearly all positive cases. However, the precision is only 0.45, meaning many false positives are likely to be predicted. The F1 score, at 0.62, suggests a moderate balance between precision and recall. Still, the model's utility in a real-world setting might be limited due to its tendency to over-predict the positive class.

Table 3. Performance comparison of DL algorithms in the detection of DDoS attacks

Model	Accuracy	Precision	Recall	F1-Score
MLP	0.48	0.99	0.45	0.62
ANN	0.73	0.63	0.88	0.74
CNN	0.48	0.45	0.81	0.58
LSTM	0.91	0.86	0.94	0.90
RNN	0.66	0.58	0.74	0.66

The ANN demonstrates a solid predictive capability, achieving an accuracy of 0.73. Its precision of 0.63 and recall of 0.88 indicate that while the model is effective in detecting actual attack instances, it does allow for a moderate level of false positives. An F1-score of 0.74 reflects a fair trade-off between precision and recall, suggesting that with further optimization, particularly to enhance precision, ANN could serve as a viable solution for DDoS detection. The CNN model has an accuracy on par with the MLP at 0.48, suggesting challenges in correctly classifying cases. The precision is low at 0.45, and the recall is 0.81, which is lower than MLP's but still indicates a tendency to identify most positive instances. The F1 score of 0.58 reflects a model that is better at ensuring attacks are not missed rather than precisely identifying only the actual attacks. For network security, this could mean a higher operational load due to false alarms.

The RNN yields a moderate accuracy of 0.66, with precision and recall values of 0.58 and 0.74, respectively. Its F1-score of 0.66 shows that the model strikes a reasonable balance between detecting attacks and minimizing incorrect classifications. However, its performance trails that of LSTM, likely due to RNN's limitations in retaining long-term dependencies—an essential factor when DDoS attack patterns span across extended traffic sequences. In contrast, the LSTM model excels in performance with an accuracy of 0.91. Its precision of 0.86 and recall of 0.94 highlight its ability to not only detect attack flows with high sensitivity but also to minimize false alarms. F1-score of 0.90 reinforces its effectiveness in handling sequential network data, making LSTM particularly well-suited for identifying sophisticated DDoS patterns in SDN environments. These results suggest that LSTM is a strong candidate for real-time deployment in intelligent network defense systems.

Figure 3 represents a performance comparison of DL algorithms in terms of accuracy and recall. Figure 4 depicts the evaluation of the models in terms of precision and F1 score. It can be concluded that DL models, especially those tailored for sequence data such as LSTMs and RNNs, have proven superior in accuracy compared to other models when handling tasks involving sequential or time-series data. This superiority stems from their architecture's adeptness at capturing temporal dependencies and contextual nuances from the data, crucial for identifying patterns indicative of DDoS attacks. Despite their ability to deliver high predictive performance, DL models introduce challenges such as heightened complexity, increased computational overhead, and often an opaque nature that complicates interpretation and integration. In real-time DDoS detection systems, where errors like false positives or false negatives carry significant consequences, the superior predictive accuracy of DL models, particularly LSTMs, renders them more suitable despite their greater resource demands. Therefore, despite being more resource-intensive and less interpretable, DL models offer enhanced predictive capabilities and are better suited for environments prioritizing top-tier performance, notwithstanding their heightened complexity and computational requirements.

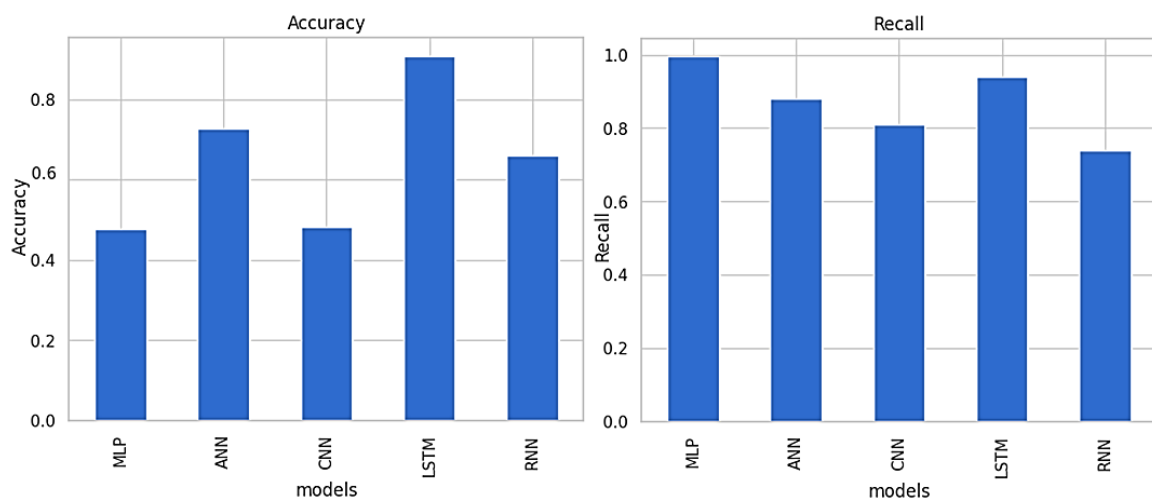


Figure 3. Performance comparison of various models concerning the accuracy and recall



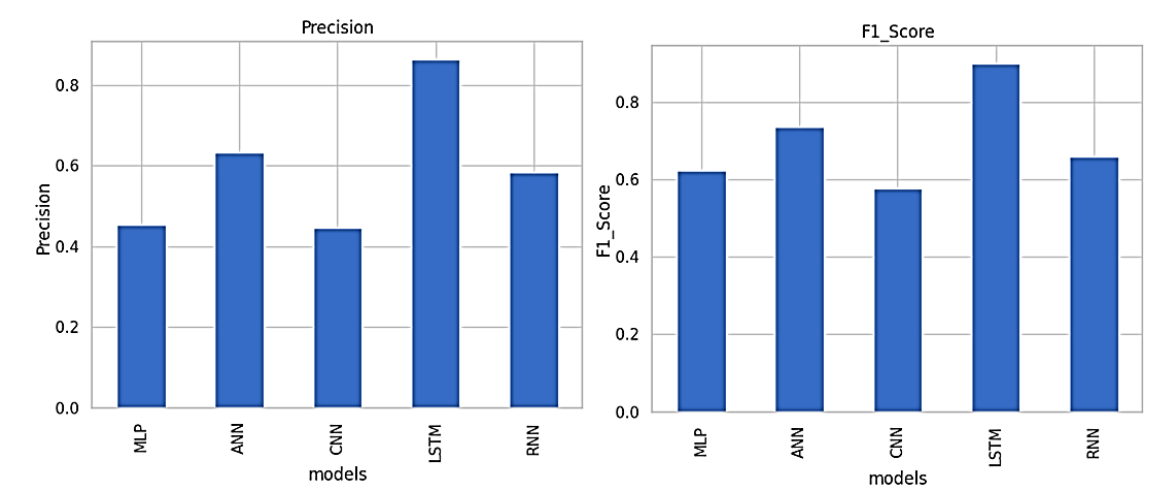


Figure 4. Performance comparison of various models concerning the precision and F1 score

4. CONCLUSION

This study explored the use of multiple prominent DL models for the detection of DDoS attacks in SDN environments. Unlike many existing works that examine only a limited selection of techniques, our approach offered a comprehensive evaluation across a diverse range of DL architectures. While the results highlight strong predictive capabilities for some models, practical deployment requires more than just high accuracy. Overall, the architecture of LSTMs emerged as particularly well-suited for handling the sequential and temporal complexities inherent in DDoS attack prediction. RNN provided a simpler yet effective alternative. ANN and MLP could be viable for less intricate or temporally dependent data. At the same time, CNN might necessitate significant adjustments or be more suitable for problems aligned with its spatial processing strengths. LSTM exhibited a high recall rate without significantly compromising precision, crucial in DDoS detection, where missing an attack can have severe consequences. The high F1 scores attained by LSTM suggest its proficiency in accurately classifying both attack and non-attack instances in a balanced manner. To ensure robustness and real-world applicability, methods such as cross-validation and testing against previously unseen data must be consistently applied; moreover, models demonstrating lower effectiveness present opportunities for improvement through refined feature selection strategies and hyperparameter tuning. Given the evolving threat landscape, integrating DL approaches into DDoS detection and mitigation strategies is no longer optional—it is essential for modern, intelligent network defense.

FUNDING INFORMATION

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Neethu S.	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
H. V. Ravish Aradhya		✓				✓		✓	✓	✓	✓	✓		
Viswavardhan Reddy	✓		✓	✓			✓			✓	✓		✓	✓
Karna														

C : Conceptualization	I : Investigation	Vi : Visualization
M : Methodology	R : Resources	Su : Supervision
So : Software	D : Data Curation	P : Project administration
Va : Validation	O : Writing - Original Draft	Fu : Funding acquisition
Fo : Formal analysis	E : Writing - Review & Editing	

## CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. Authors state no conflict of interest.





## DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, [NS], upon reasonable request.





## REFERENCES

- [1] B. Alhijawi, S. Almajali, H. Elgala, H. B. Salameh, and M. Ayyash, "A survey on DoS/DDoS mitigation techniques in SDNs: classification, comparison, solutions, testing tools and datasets," *Computers and Electrical Engineering*, vol. 99, 2022, doi: 10.1016/j.compeleceng.2022.107706.
- [2] J. Singh and S. Behal, "Detection and mitigation of DDoS attacks in SDN: a comprehensive review, research challenges and future directions," *Computer Science Review*, vol. 37, 2020, doi: 10.1016/j.cosrev.2020.100279.
- [3] V. Hnamte, A. A. Najar, H. N.-Nguyen, J. Hussain, and M. N. Sugali, "DDoS attack detection and mitigation using deep neural network in SDN environment," *Computers & Security*, vol. 138, 2024, doi: 10.1016/j.cose.2023.103661.
- [4] R. B. Said, Z. Sabir, and I. Askerzade, "CNN-BiLSTM: A hybrid deep learning approach for network intrusion detection system in software-defined networking with hybrid feature selection," *IEEE Access*, vol. 11, pp. 138732–138747, 2023, doi: 10.1109/ACCESS.2023.3340142.
- [5] H. S. Ilango, M. Ma, and R. Su, "A feed forward-convolutional neural network to detect low-rate DoS in IoT," *Engineering Applications of Artificial Intelligence*, vol. 114, 2022, doi: 10.1016/j.engappai.2022.105059.
- [6] G. D. L. T. Parra, P. Rad, K.-K. R. Choo, and N. Beebe, "Detecting internet of things attacks using distributed deep learning," *Journal of Network and Computer Applications*, vol. 163, 2020, doi: 10.1016/j.jnca.2020.102662.
- [7] M. A. Ribeiro, M. S. P. Fonseca, and J. de Santi, "Detecting and mitigating DDoS attacks with moving target defense approach based on automated flow classification in SDN networks," *Computers & Security*, vol. 134, 2023, doi: 10.1016/j.cose.2023.103462.
- [8] O. Habibi, M. Chemmakha, and M. Lazaar, "Imbalanced tabular data modelization using CTGAN and machine learning to improve IoT bot-net attacks detection," *Engineering Applications of Artificial Intelligence*, vol. 118, 2023, doi: 10.1016/j.engappai.2022.105669.
- [9] N. S. and H. V. R. Aradhya, "Detection of (DDoS) attacks in SDN," *ECS Transactions*, vol. 107, no. 1, pp. 18305–18313, Apr. 2022, doi: 10.1149/10701.12189ecst.
- [10] M. A. Alsoufi *et al.*, "Anomaly-based intrusion detection model using deep learning for IoT networks," *Computer Modeling in Engineering & Sciences*, vol. 141, no. 1, pp. 1–10, Aug. 2024, doi: 10.32604/cmescs.2024.052112.
- [11] E.-B. Donkol, A. G. Hafez, A. I. Hussein, and M. M. Mabrook, "Optimization of intrusion detection using likely point PSO and enhanced LSTM-RNN hybrid technique in communication networks," *IEEE Access*, vol. 11, pp. 9469–9482, 2023, doi: 10.1109/ACCESS.2023.3240109.
- [12] H. Zhou, Y. Zheng, X. Jia, and J. Shu, "Collaborative prediction and detection of DDoS attacks in edge computing: a deep learning-based approach with distributed SDN," *Computer Networks*, vol. 225, 2023, doi: 10.1016/j.comnet.2023.109642.
- [13] Y. Liu, T. Zhi, M. Shen, L. Wang, Y. Li, and M. Wan, "Software-defined DDoS detection with information entropy analysis and optimized deep learning," *Future Generation Computer Systems*, vol. 129, pp. 99–114, Apr. 2022, doi: 10.1016/j.future.2021.11.009.
- [14] K. N. Rao, K. V. Rao, and P. V. G. D. P. Reddy, "A hybrid intrusion detection system based on sparse autoencoder and deep neural network," *Computer Communications*, vol. 180, pp. 77–88, 2021, doi: 10.1016/j.comcom.2021.08.026.
- [15] N. M. Y.-Naula, C. V.-Rosales, J. A. P.-Díaz, and D. F. Carrera, "A flexible SDN-based framework for slow-rate DDoS attack mitigation by using deep reinforcement learning," *Journal of Network and Computer Applications*, vol. 205, 2022, doi: 10.1016/j.jnca.2022.103444.
- [16] H. Polat, M. Türkoğlu, O. Polat, and A. Şengür, "A novel approach for accurate detection of the DDoS attacks in SDN-based SCADA systems based on deep recurrent neural networks," *Expert Systems with Applications*, vol. 197, 2022, doi: 10.1016/j.eswa.2022.116748.
- [17] V. R. Karna and K. V. V. Reddy, "1-dimensional convolutional neural networks for predicting sudden cardiac," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 13, no. 1, pp. 984–993, 2023, doi: 10.11591/ijai.v13.i1.pp984-993.
- [18] Z. Xu, "Deep learning based DDoS attack detection," *ITM Web of Conferences*, vol. 70, 2025, doi: 10.1051/itmconf/20257003005.
- [19] Y. Cao *et al.*, "Detecting and mitigating DDoS attacks in SDN using spatial-temporal graph convolutional network," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 6, pp. 3855–3872, 2022, doi: 10.1109/TDSC.2021.3108782.
- [20] P. V. Shalini, V. Radha, and S. G. Sanjeevi, "Early detection and mitigation of TCP SYN flood attacks in SDN using chi-square test," *Journal of Supercomputing*, vol. 79, pp. 10353–10385, 2023, doi: 10.1007/s11227-023-05057-x.
- [21] V. Hnamte, A. A. Najar, H. N.-Nguyen, J. Hussain, and M. Naik, "DDoS attack detection and mitigation using deep neural network in SDN environment," *Computers & Security*, vol. 138, 2024, doi: 10.1016/j.cose.2023.103661.
- [22] T. H. H. Aldhyani and H. Alkahtani, "Cyber security for detecting distributed denial of service attacks in agriculture 4.0: deep learning model," *Mathematics*, vol. 11, no. 1, 2023, doi: 10.3390/math11010233.
- [23] E. Mushtaq, A. Zameer, and A. Khan, "A two-stage stacked ensemble intrusion detection system using five base classifiers and MLP with optimal feature selection," *Microprocessors and Microsystems*, vol. 94, Oct. 2022, doi: 10.1016/j.micpro.2022.104660.
- [24] P. L. S. Jayalaxmi *et al.*, "DeBot: A deep learning-based model for bot detection in industrial internet-of-things," *Computers and Electrical Engineering*, vol. 102, 2022, doi: 10.1016/j.compeleceng.2022.108214.
- [25] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, "Anomaly-based intrusion detection system for IoT networks through deep learning model," *Computers and Electrical Engineering*, vol. 99, 2022, doi: 10.1016/j.compeleceng.2022.107810.





**BIOGRAPHIES OF AUTHORS**

**Neethu S.**     received the B.Tech. degree in Electronics and Communication Engineering from Calicut University, India, in 2009, and the M.Tech. degree in Embedded Systems from Amrita Vishwa Vidyapeetham University, India, in 2012. She is currently pursuing a Ph.D. degree at RV College of Engineering under Visvesvaraya Technological University, India. She is working as an Assistant Professor at RV College of Engineering. Her research interests include network security, software-defined networking, machine learning, and wireless communication. She can be contacted at email: [neethus@rvce.edu.in](mailto:neethus@rvce.edu.in).



**Dr. H. V. Ravish Aradhya**     is currently Head of the Department of Electronics and Communication Engineering at RV College of Engineering. He has academic and research experience of over 30 years. He has guided over 80 projects for undergraduate and postgraduate students. He has guided 2 Ph.D. scholars and is currently guiding four other scholars. He has over 47 journal publications and 75 internal conference publications, with a good number of citations serving the research community; Scopus (400+), WoS (90+), Semantic Scholar (300+), and Google (650+) citations. He holds a B.E. (Electronics) and M.E (Electronics) from Bangalore University and a Ph.D. (Engineering) from VTU, Belagavi. He can be contacted at email: [ravisharadhya@rvce.edu.in](mailto:ravisharadhya@rvce.edu.in).



**Dr. Viswavardhan Reddy Karna**     received the Bachelor of Technology degree in Electronics and Communication Engineering from JNTU-Hyderabad, A. P., India, in 2008, and the M.S. degree in Telecommunication Systems from Blekinge Institute of Technology, Karlskrona, Sweden, in 2011. Further, received a Doctor of Philosophy (Electronics Engineering) from Jain University, Bengaluru, India in 2023. Currently, he is working as an Assistant Professor in the Department of AIML, RV College of Engineering, Bengaluru, India. His research interests include machine learning, artificial intelligence, wireless body area networks, wireless sensor networks, and software-defined networks. He can be contacted at email: [viswavardhank@rvce.edu.in](mailto:viswavardhank@rvce.edu.in).