# Mô hình học sâu Long short-Term memory phát hiện tấn công DDos

## TÓM TẮT

Gần đây, các mối đe dọa tấn công Từ chối dịch vụ phân tán-Distributed Denial of Service (DDoS) đang trở nên phức tạp, tinh vi, gây ra thách thức cho các hệ thống bảo vệ thông thường. Việc phát hiện sớm các dấu hiệu tấn công rất quan trọng, để bảo vệ và chống lại các mối đe dọa tấn công. Nghiên cứu đề xuất sử dụng mô hình dựa trên kỹ thuật Học sâu Mạng bộ nhớ Dài-Ngắn - Long Short-Term Memory (LSTM). Kỹ thuật này gồm một số thuật toán lựa chọn và trích xuất đặc trưng, được tự động cập nhật trong quá trình huấn luyện. Với số lượng dữ liệu nhỏ, LSTM vẫn hoạt động nhanh và chính xác. Nghiên cứu đã tiến hành thử nghiệm trên tập dữ liệu CICDDoS2019 và kết quả cho thấy mô hình đạt được các chỉ số hiệu suất như sau: Độ chính xác (Accuracy) đạt 95%, Precision đạt 96%, độ phủ (Recall) đạt 93% và điểm F1 (F1 Score) là 95%. Mục tiêu của nghiên cứu, đưa ra được một mô hình có khả năng xử lý dữ liệu chuỗi và lưu trữ thông tin học được lâu dài. Có thể tích hợp mô hình vào các hệ thống giám sát và bảo mật mạng, cải thiện khả năng phát hiện phản ứng với các mối đe dọa tấn công mạng ngày càng phức tạp.

**Từ khóa:** *DDoS, DoS, LSTM, Học sâu, Học máy*.

# Deep learning model using Long short-Term memory for DDoS attack detection

**ABSTRACT**

Recently, Distributed Denial of Service (DDoS) attack threats have become increasingly complex and sophisticated, posing challenges to traditional protection systems. Early detection of attack signs is crucial for safeguarding against these threats. This study proposes the use of a model based on Long Short-Term Memory (LSTM) deep learning techniques. This approach includes several feature selection and extraction algorithms that are automatically updated during training. Even with a small amount of data, LSTM performs quickly and accurately. Experiments were conducted on the CICDDoS2019 dataset, and results show that the model achieved performance metrics as follows: Accuracy of 95%, Precision of 96%, Recall of 93%, and F1 Score of 95%. The goal of the research is to develop a model capable of processing sequential data and retaining learned information over the long term. This model can be integrated into network monitoring and security systems, enhancing the ability to detect and respond to increasingly complex cyberattack threats.

**Keyword:** *DDoS, DoS, LSTM, Deep Learning, Machine Learning*

## 1. INTRODUCTION

The rapid development of services through the Internet, such as financial and banking transactions, communication, e-commerce, online shopping, online payments, healthcare, and education, presents significant challenges for user safety. There are currently numerous methods of network attacks aimed at disruption, among which the two fundamental attack methods are Denial of Service (DoS) attacks, which aim to prevent legitimate users from accessing network resources, and Distributed Denial of Service (DDoS) attacks, which represent a more advanced form of DoS attacks. The primary difference between DoS and DDoS attacks lies in the scope of the attack. DoS attack traffic primarily originates from one or a few source hosts, whereas DDoS attack traffic originates from a large number of hosts dispersed across the Internet.

There are two DDoS attack methods. The first involves the attacker sending specially crafted packets that cause errors in the transmission protocol or the application running on the victim's machine; a typical example of this method is exploiting security vulnerabilities in the protocols or services on the victim's machine. The second, more common DDoS attack method can be categorized into two forms: the first form disrupts the user's connection to the service server by flooding the network transmission or exhausting bandwidth and network resources. The second form disrupts the service provided to the user by depleting the resources of the service server, such as CPU processing time, memory, disk bandwidth, and database capacity. This type of attack includes application-layer flooding attacks. These attacks are expected to become increasingly complex as Internet technologies evolve. Traditional methods for assessing the risks of DDoS attacks often have low accuracy and slow response times. To address this issue, cybersecurity experts and scholars are researching the use of Machine Learning (ML) and Deep Learning (DL) techniques for DDoS detection [1]. ML and DL methods hold tremendous potential for detecting network attacks due to their ability to classify more accurately and effectively [2]. Currently, methods such as Random Forest (RF), K-Nearest Neighbors (KNN), and Naive Bayes are in use. For deep learning, techniques such as Artificial Neural Networks (ANN), Deep Neural Networks (DNN), and Recurrent Neural Networks (RNN) are commonly employed [3]. After surveying and evaluating multiple methods, this research decided to implement the Long Short-Term Memory (LSTM) deep learning architecture. The LSTM architecture can capture information based on temporal correlation, which is a critical characteristic of DDoS traffic, thus enabling effective prediction of network traffic. The goal of this paper is to construct an LSTM deep learning model that achieves higher accuracy than existing machine learning techniques in classifying and forecasting DDoS attacks.

In the field of network security, numerous studies have utilized machine learning techniques to classify DDoS threats, including algorithms such as K-Nearest Neighbors (KNN), Logistic Regression, Random Forest (RF), Support Vector Machine (SVM), and Naive Bayes classification [4]. KNN identifies the K nearest neighbors of the input data. A voting technique is used to classify test data. Several studies have employed KNN to classify DDoS attacks with quite effective results. Research using SVM has distinguished between normal and malicious network traffic based on IP address interaction features [5]. This model demonstrated good discrimination between malicious and normal traffic flows. Zheng et al. utilized RF to classify DDoS attacks, achieving classification performance with a precise feature set [6]. Naive Bayes, a classification algorithm based on Bayes' theorem, has been effectively used for classifying DDoS attacks when features are independent. Riadi applied the Naive Bayes method to detect DDoS attacks using mean and standard deviation, achieving favorable results. Ahanger developed an Artificial Neural Network (ANN) model to detect DDoS network attacks, obtaining certain results [7]. Zubair Hasan, Sattar, and Zahid Hasan conducted research on using a Deep Convolutional Neural Network (DCNN) model to detect and mitigate DDoS attacks [8]. Given that network traffic is often vast and data-rich, traditional machine learning algorithms can struggle with analysis. Consequently, DCNN has proven an effective solution in this case. Research results indicated that DCNN outperformed traditional machine learning algorithms such as Naive Bayes, SVM, and KNN, with accuracies of 80%, 87%, and 92%, respectively. This marks a significant advancement in processing and protecting networks against increasingly sophisticated and complex DDoS attacks [9].

According to research by Zhu, Ye, and Xu, deep learning models like Feedforward Neural Networks (FNN) and Convolutional Neural Networks (CNN) have proven effective in analyzing network traffic and detecting DDoS attacks [10]. Experiments on the NSL-KDD dataset demonstrated that deep learning models achieved superior accuracy in detecting anomalies and network incidents compared to other machine learning techniques such as RF, Logistic Regression, and SVM. This indicates that utilizing deep learning techniques is a viable option for enhancing network security. In monitoring unusual activities on networks via Intrusion Detection Systems (IDS), Alzahrani and Hong proposed using Artificial Neural Network models [11].

Recently, research on the CICDDoS2019 sample dataset for predicting DDoS attacks has yielded quite high accuracy results. Each research effort applied a unique technique with its strengths. In the study by Ahuja et al. [12], the prediction accuracy achieved was 95.6%.

## 2. RESEARCH METHODOLOGY

Recurrent Neural Networks (RNNs) are a type of neural network designed to process sequential data. The main idea behind RNNs is to use a memory mechanism to retain information from previous computation steps, enabling accurate predictions for subsequent steps. The architecture of an RNN can be represented as a series of recurrent units. Each unit connects to the previous unit, forming a directed cycle. At each time step, the recurrent unit receives the current input, combines it with the hidden state from the previous step, generates an output, and updates the hidden state for the next time step. This process repeats for each input in the sequence, allowing the RNN to gather information about relationships and patterns over time.

The computation process of an RNN at three consecutive time steps involves the following: at time step t, the current input $X_t$ is concatenated with the hidden state $H_{t-1}$ from the previous time step, which is then fed into a fully connected layer with an activation function $\emptyset$. The output of this layer is the hidden state $H_t$ at time step t, which also serves as input for the output layer $O_t$. The model parameters $W_{xh}$ and $W_{hh}$, along with $H_t$, are used to compute the hidden state $H_{t+1}$ at the next time step t+1. This process allows the RNN to understand and retain important information across time steps, making it suitable for tasks that require processing sequential data, such as time series prediction. The architectural model of the RNN is illustrated in Figure 1.

.

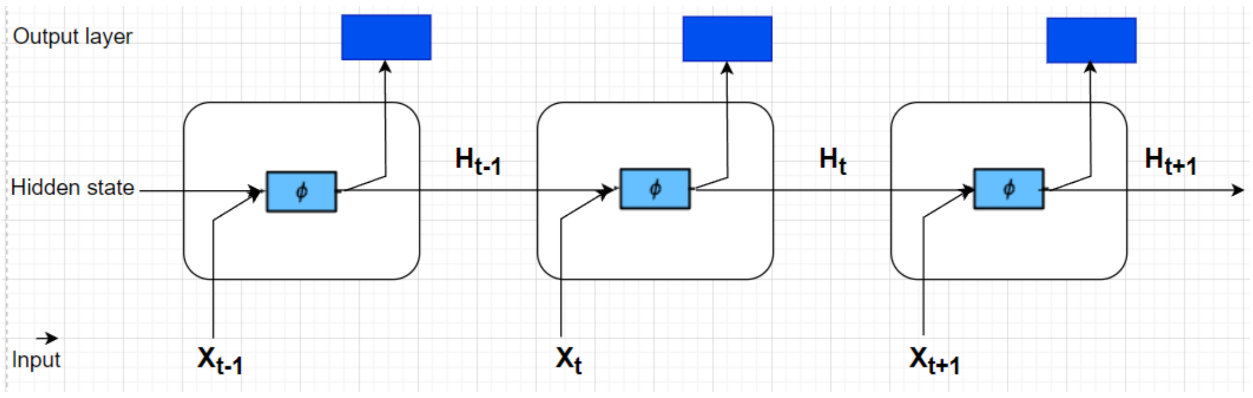

**Figure 1.** RNN Architecture.

To manage data and feed it into the model, the LSTM model will utilize three gates known as the forget gate, input gate, and output gate. These gates are the core components of the LSTM model and are responsible for the overall control of the model. The forget gate determines which parts of the old information should be discarded based on the previous hidden state and the current input value. The input gate decides what new data should be allowed into the network based on relevant information that should be introduced to the LSTM's memory state, referred to as the cell state. The output gate determines the new hidden state based on the updated cell state and the current input value.

The computational formulas for the gates are as follows:

$$F_{(t)} = \sigma(w_f[h_{(t-1)}, X_t] + b_f) \quad (1)$$

$$I_{(t)} = \sigma(w_i\,[[h_{(t-1)}, X_t] + b_i]) \quad (2)$$

$$O_{(t)} = \sigma(W_0 * [h_{(t-1)}, X_t] + b_0) \quad (3)$$

$$f(x) = \frac{1}{1^{axt}} \quad (4)$$

$$tanh(x) = \frac{2}{1+e^{-2x}} - 1 \quad (5)$$

$$\widehat{C_t} = \tanh(w_c\,, [h_{t-1}, X_t] + b_c) \quad (6)$$

$$c_t = F_t . C_{(t-1)} + I_t . \widehat{C_t} \quad (7)$$

$$h_t = O_t . tanh(C_t) \quad (8)$$

Where: $h_{(t-1)}$ is the previous hidden state, $x_t$ is the current input value, $F_{(t)}$, $I_{(t)}$ and $O_{(t)}$ are the values of the forget, input, and output gates, respectively, and W and b are the weights and biases, respectively (1). The sigmoid function sigmoid (σ) is used to extract information from both the most recent input and the previous hidden layer. The range of the sigmoid function sigmoid varies from 0 to 1, while the range of the tanh function varies from -1 to 1. If the value of the sigmoid function is close to 1, it retains the data; if it is close to 0, it discards the data. The architectural model of LSTM is illustrated in the Figure2.

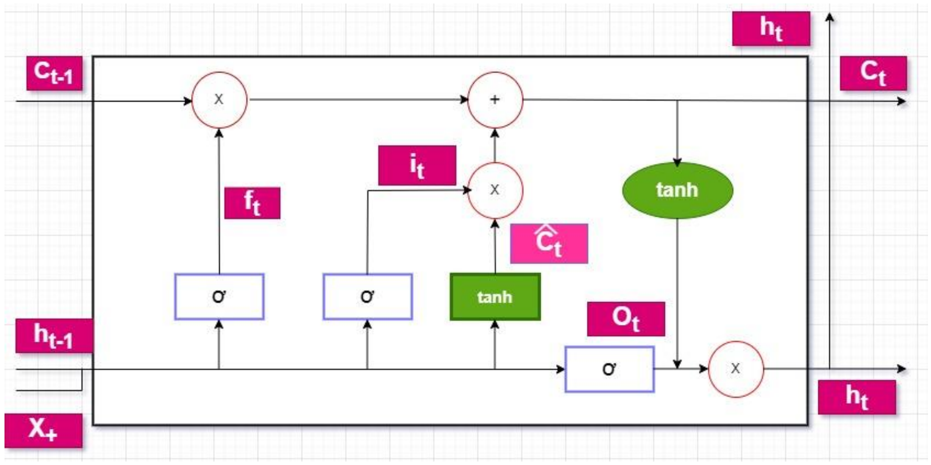

**Figure 2**. LSTM Architecture.

## 3. RESULTS AND DISCUSSION

### 3.1. Experimental Environment

This experiment was conducted using Python version 3.9, along with machine learning libraries such as TensorFlow and Scikit-learn, and other supporting libraries. The computer used for this experiment was equipped with an 8-core processor and 16 threads, along with 32GB of RAM. Additionally, a GPU was utilized to accelerate computation, specifically the NVIDIA GeForce RTX 3080.

### 3.2. Dataset

This study employed the CICDDoS2019 sample dataset. The "DDoS Evaluation Dataset (CICDDoS2019)" was published by the Canadian Institute for Cybersecurity on October 31, 2019, and contains 225,745 samples across 85 columns in CSV format. This dataset provides a diverse set of information about various types of network traffic and attack manifestations [13]. Many studies have been conducted on this dataset with different exploitation aspects, including some machine learning models like Convolutional Neural Networks (CNNs) [14], achieving certain results. However, since DDoS attack data is in the form of continuous time series, applying CNN models may not be effective. This model is not optimized for sequential data and often loses temporal order information due to pooling operations. In particular, CNNs do not capture long-term correlations well in time series data. Furthermore, when using CNNs for time series, preprocessing data to convert it into an appropriate format can be complex and time-consuming.

The classification process identified 128,027 samples belonging to the DDoS group and 97,718 samples belonging to the BENIGN group. The distribution of labels in the dataset is relatively balanced between the two groups, with approximately 43.3% of samples identified as benign and 56.7% identified as DDoS attacks.

Balancing the labels in the dataset is an important factor when training the model. This ensures that the model not only understands network attacks but also effectively differentiates them from legitimate activities. In practice, there are many factors affecting the model's performance, leading to prediction processes that may not meet expectations.

### 3.3. Data Visualization

Visualizing the changes in various features over time in the dataset helps identify attack patterns through normal and abnormal traffic. By doing so, one can observe the changes in variables and gain a better understanding of trends and cycles in the real-time data. The variation of features over time can provide crucial information during data analysis and model training. The data at each timestamp in the DDoS SDN dataset was visualized by grouping the data by timestamp and counting the number of rows in each group, displaying it as a chart to observe the distribution of data over time and recognize special manifestations within the data.

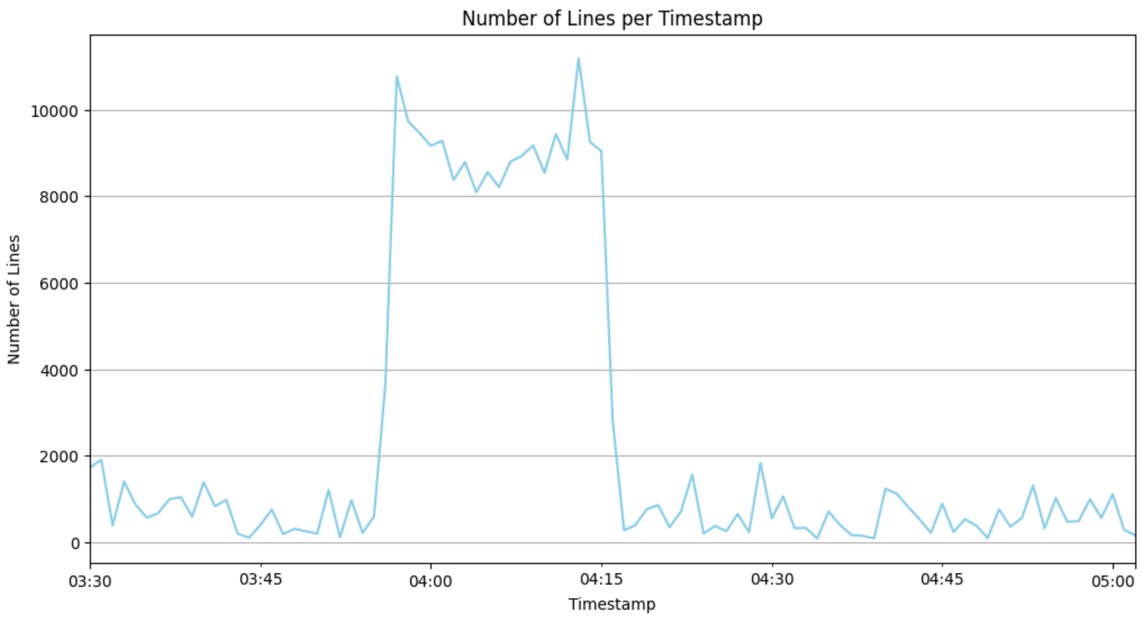

**Figure 4**. Chart Representing Data Over Time.

In addition, to compare the number of connections from DDoS attacks and non-DDoS connections (Benign) over time in the dataset, the rows labeled “BENIGN” were grouped by timestamp, and the counts in each group were displayed as a chart. This allows for the observation of fluctuations between attacks and non-attack activities within the system. The goal is to support early detection of attacks and assess their impact in real time.

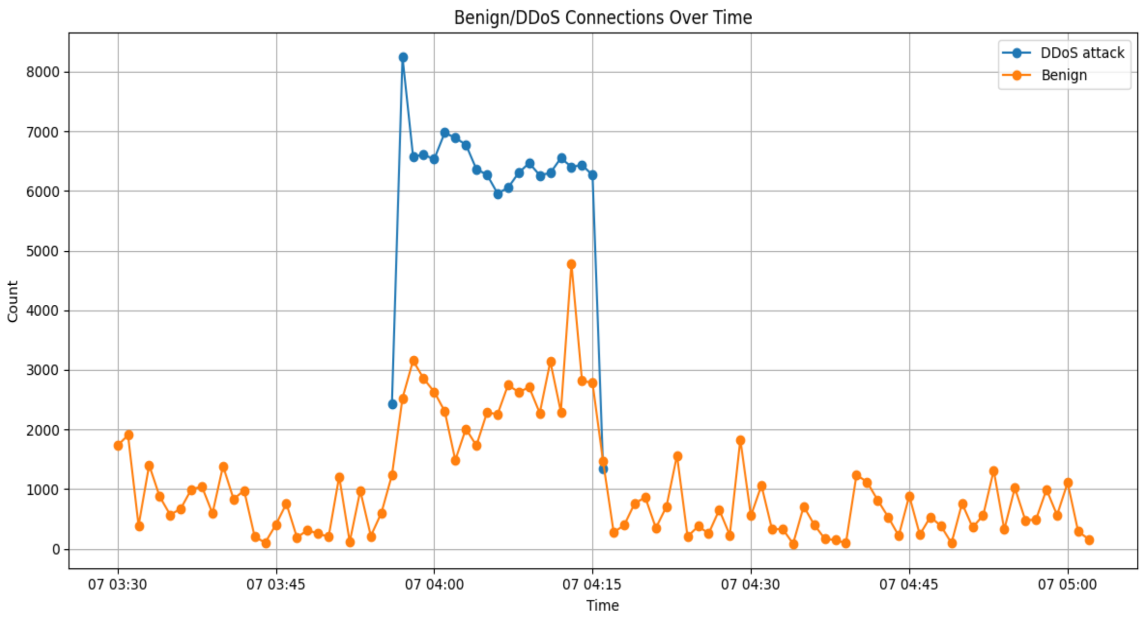

**Figure 5.** Chart Showing the Number of Attack and Normal Connections Over Time

During the data analysis process, a series of tasks were undertaken to further explore the characteristics and relationships among them in the DDoS SDN dataset. First, the label distribution of the classification features was identified to gain a better understanding of their distribution within the dataset. The next step involved analyzing and interpreting TCP flags such as PSH, FIN, SYN, RST, ACK, URG, and ECE, in order to understand their messages and impacts on the network. Finally, variables were visualized grouped by labels in the data, helping to identify the distribution and relationships between the labels in the dataset. This provided detailed and clear information about the dataset, supporting the analysis process and aiding in decision-making during system implementation as well as network security and management.

.

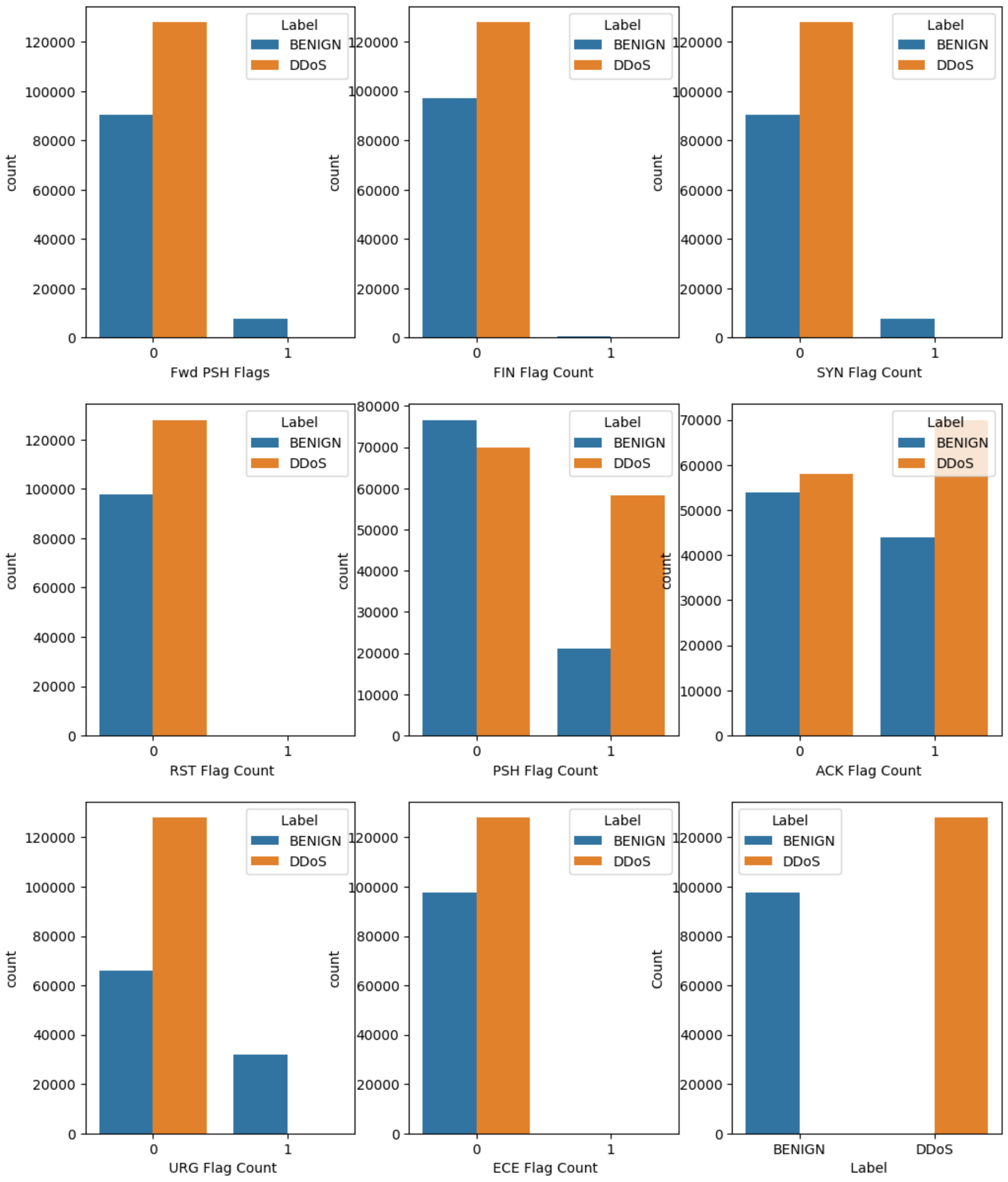

**Figure 6.** Chart Showing Grouped Labels in the Dataset

### 3.4. Data Normalization

The process of cleaning and transforming data is a crucial part of data preparation. First, constant features were removed, as these do not provide useful information since they do not change across data samples. This step helps reduce the size of the dataset and eliminate unnecessary noise.

Next, redundant features—those that can be inferred from other features—were removed. This helps lower computational costs and prevents the model from becoming overly complex. In this dataset, constant features such as 'Flow ID', 'Destination IP', 'Source IP', 'Destination Port', and 'Source Port' were eliminated. Since the Flow ID is unique, and during an attack, these IPs often operate anonymously or as fake IPs, the IP and Port fields needed to be removed. Observing the 'Protocol' field, it contained three protocols: "UDP, TCP, POP3", with TCP accounting for 85.5%, while the other two protocols contributed little and did not aid in predicting the target variable. As a result, the dataset, after refinement, consisted of 225,745 rows and 73 columns.

The next step involved label encoding for the categorical features using LabelEncoder to convert the labels into binary format. After that, a correlation analysis was performed among the features in the dataset to understand their relationships.
Finally, feature selection was conducted by removing columns with high correlations to minimize multicollinearity and redundancy in the dataset. This process was repeated to examine the correlations between the features and the target variable, aiming to retain the most meaningful and independent features. This helps refine the feature set and improve the prediction performance of the model.

The LSTM model was trained on the normalized data. The dataset was divided into training data (trainX) and corresponding labels (trainY) to adjust the model weights. The model was trained over 100 epochs. To manage training on small batches of data, the data was split into batches of 1024 samples each. To evaluate the model's performance and prevent overfitting, 20% of the training data was used as validation data. The training process was displayed for each epoch, allowing for tracking the model's progress and understanding the variations of metrics such as loss and accuracy over iterations. Finally, information about the training process was stored in the variable historylog, which included the metric values during training and validation after each epoch. This aids in analyzing and evaluating the model's performance once training is complete. The processes of data collection, processing, and model training are illustrated in the overall model as shown in Figure 7.

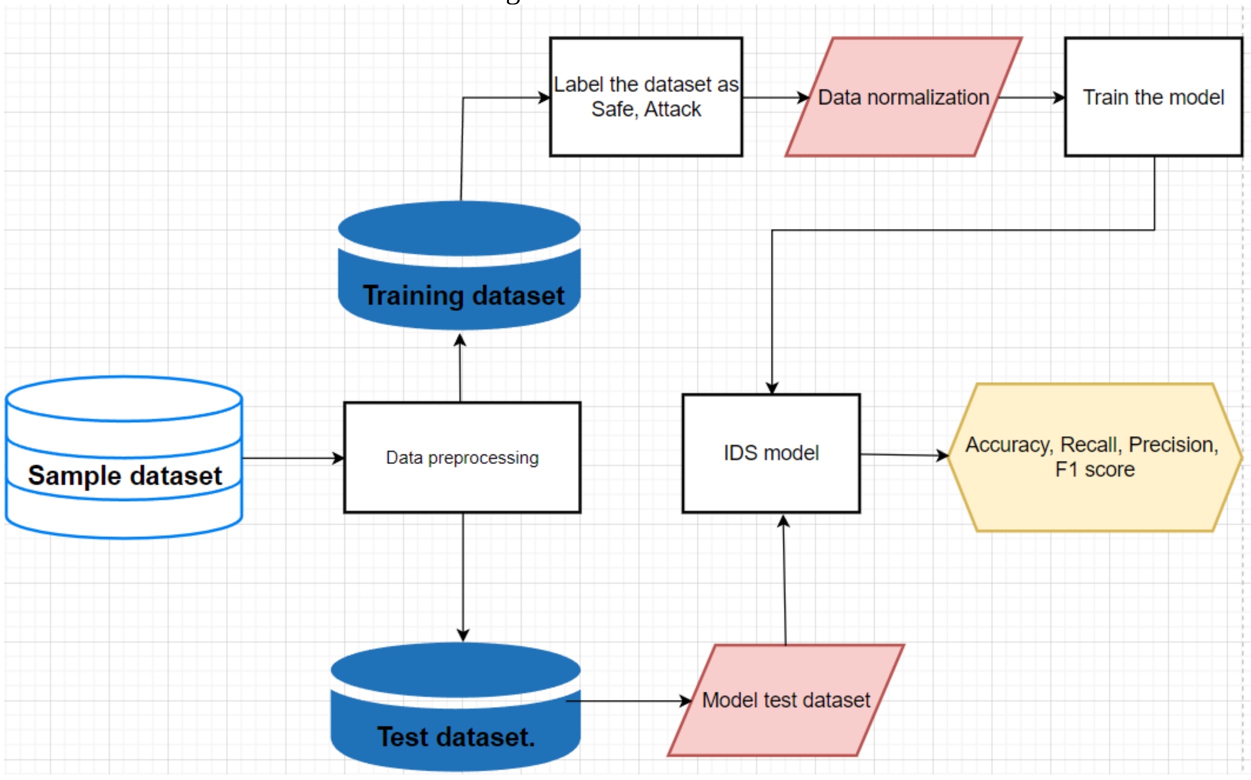

**Figure 7**. Proposed LSTM model

### 3.5. Performance Evaluation

The initial loss during training was quite high but gradually decreased over the epochs, indicating that the model was learning. The validation loss also decreased, although at a lower rate, suggesting that the model generalized reasonably well to new, unseen data not used during training

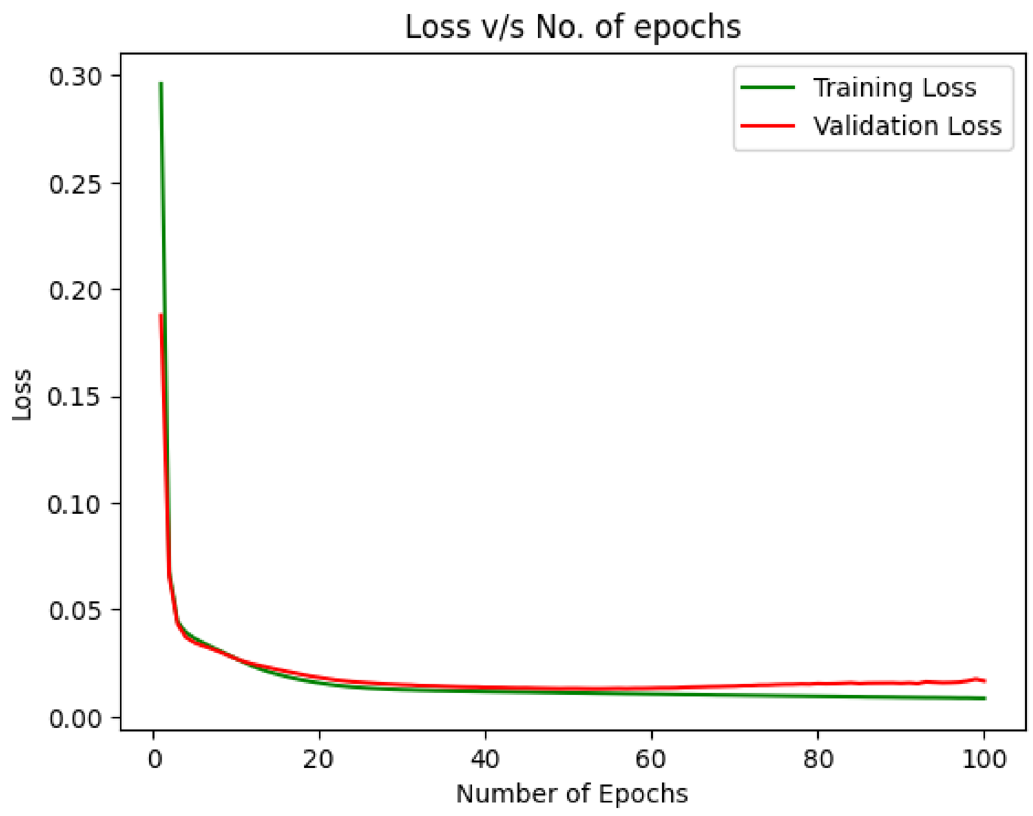

**Figure 8**. Representation of the Model's Loss Value

In addition, the accuracy during the initial learning phase was average, but it quickly increased to reach an optimal level. The validation accuracy followed a similar trend, although it improved at a slower rate compared to the training process

.

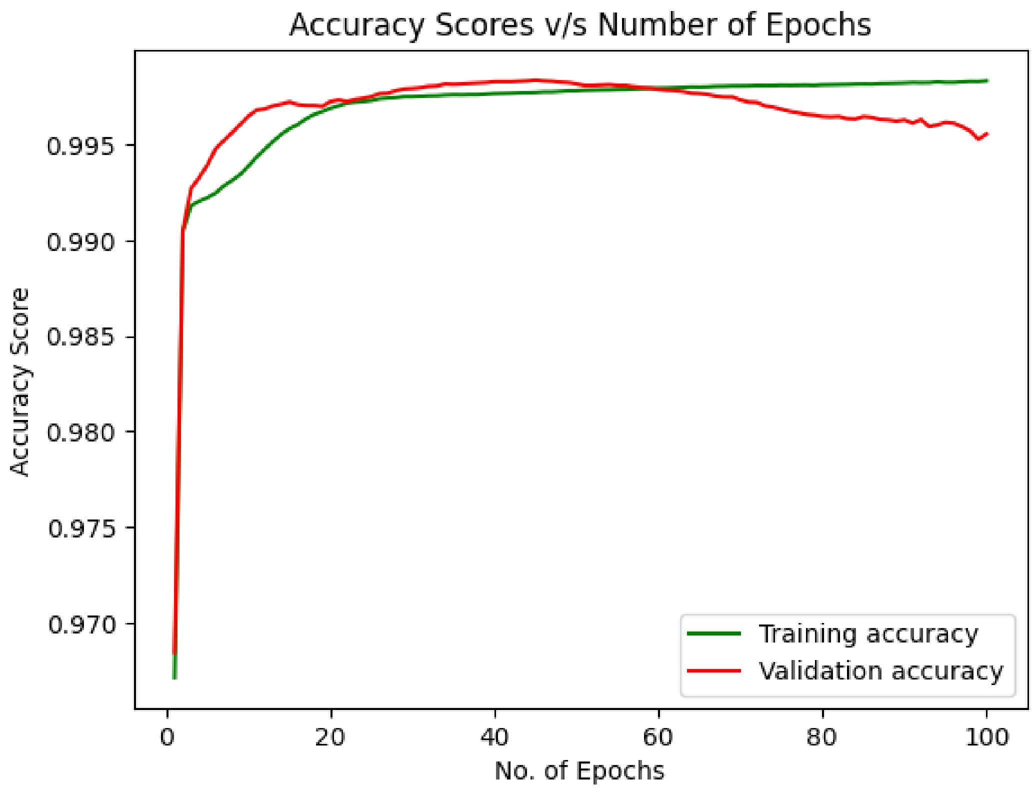

**Figure 9.** Representation of the Accuracy Score

To evaluate the performance of machine learning and deep learning algorithms, selecting appropriate evaluation criteria is crucial. This study uses the metrics of Accuracy, Precision, Recall, and F1-score to assess the model's performance. Formula (9) calculates accuracy, with values ranging from 0 to 1:

Precision measures the ratio of correctly predicted cases among those predicted as positive. It is calculated by dividing the number of true positives by the total number of cases predicted as positive (true positives plus false positives).

According to the formula:

$$\text{Precision} = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)} \quad (9)$$

Recall measures the ratio of correctly predicted cases among all actual positive cases. It is calculated by dividing the number of true positives by the total number of actual positive cases (true positives plus false negatives), as given by the formula:

$$\text{Recall} = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative (FN)} \quad (10)$$

Accuracy measures the ratio of correct predictions (including both positive and negative predictions) to the total number of data points. It is calculated using the formula:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + PN} \quad (11)$$

The F1 score is a harmonic mean of Recall and Precision, calculated using the following formula:

$$\text{F1 Score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (12)$$

Based on these performance evaluation metrics, the quality of the proposed model can be assessed, allowing for appropriate decisions to improve the model in the future

**3.6. Results**

This study focuses on evaluating the performance of the Long Short Term Memory (LSTM) model using the CICDDoS2019 dataset. This dataset is divided into two parts: one part is used to train the model, and the other is reserved for testing. Performance metrics such as Accuracy, Recall, F1-score, and Precision are used for comparison.

The CICDDoS2019 dataset has been the subject of various studies with high accuracy; however, each technique has its unique characteristics. The Optimized MLP-CNN Model [15] is suitable for detecting complex patterns from non-sequential data. While CNNs excel at spatial feature extraction, they are not as effective in handling time series data.

The LSTM model is better suited for scenarios requiring time series analysis, especially for DDoS attacks that exhibit prolonged or repetitive characteristics. The results of the model's performance evaluation on the test set are presented in Table 1, providing an overview of the model's predictive capability in real-world conditions. This plays a crucial role in the assessment.

**Table 1.** Performance Results of the Model on the Test Dataset

| Reference | **Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-score** | **Method** |
|---|---|---|---|---|---|---|
| Long Short-Term Memory Model | CICDDoS2019 | 0,95 | 0,96 | 0,93 | 0,95 | LSTM |
| Ahuja et al [12] | CICDDoS2019 | 0,95 | 0,96 | 0,92 | 0,94 | CNN |
| Thakkar, A. and Lohiya, R [15] | CICDDoS2019 | 99,95 | 99,90 | 99,98 | 99,79 | MLP-CN |

## 4. CONCLUSION

Timely detection of remote DDoS attacks is crucial for ensuring cybersecurity and protecting personal information. This study proposes the use of Long Short Term Memory (LSTM) deep learning model for forecasting DDoS attacks, with experimental results demonstrating its superior effectiveness compared to traditional machine learning methods. The LSTM model has proven flexible in feature selection and extraction, achieving an accuracy of up to 93%

in classifying DDoS attacks. This significantly surpasses other traditional models when trained and evaluated on the same dataset. Utilizing the LSTM model for DDoS attack detection can serve as a foundation for research on high-accuracy intrusion detection. Integrating the

LSTM model into software-defined networks could be a beneficial choice, and further research on reinforcement learning using updated traffic patterns can adapt to new types of attacks. This proposal offers a novel solution to current network security challenges, effectively detecting and responding to DDoS attacks, supported by the proven high performance of the LSTM model.

## REFERENCES

[1] D. Kumar, R.K. Pateriya, R.K. Gupta, V. Dehalwar, and A. Sharma, “Computer Science and Engineering Department, Maulana Azad National Institute of Technology, Bhopal, India, 462003. Computer Science and Engineering Department, Pandit Deendayal Energy University, Gandhinagar, India, 382007. Department of Computer Science & Engineering, University of Petroleum & Energy Studies, Dehradun, 248007, India.”

[2] A. Huynh et al., “A method of Deep Reinforcement Learning for Simulated Autonomous Vehicle Control,” in Proc. 16th Int. Conf. Evaluation Novel Approaches Softw. Eng. (ENASE), 2021, pp. 372–379.

[3] D. Dave, M. Kava, R.K. Gupta, and K. Shah, “Deep Learning approach for Intrusion Detection System,” in IEEE Int. Conf. Technol., Res., Innov. Betterment Soc. (TRIBES), 2022, doi: 10.1109/TRIBES52498.2021.9751643.

[4] U. Dincalp, “Anomaly based distributed denial of service attack detection and prevention with machine learning,” in Proc. 2nd Int. Symp. Multidiscip. Stud. Innov. Technol., Ankara, Turkey, Oct. 19–21, 2018.

[5] A. Fadlil, I. Riadi, and S. Aji, “Review of detection DDoS attack detection using Naïve Bayes classifier for network forensics,” Bull. Electr. Eng. Inform., vol. 6, pp. 140–148, 2017. [CrossRef].

[6] C. Wang, J. Zheng, and X. Li, “Research on DDoS attacks detection based on RDF-SVM,” in Proc. 10th Int. Conf. Intell. Comput. Technol. Autom., Changsha, China, Oct. 9–12, 2017.

[7] T.A. Ahanger, “An effective approach of detecting DDoS using artificial neural networks,” in Proc. 2017 Int. Conf. Wireless Commun., Signal Process. Netw., Chennai, India, Mar. 22–24, 2017, pp. 707–711.

[8] Md. Z. Hasan, K.M.Z. Hasan, and A. Sattar, “Burst header packet flood detection in optical burst switching network using deep learning model,” Procedia Comput. Sci., vol. 143, pp. 970–977, 2018.

[9] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, and D. Siracusa, “Lucid: A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection,” IEEE Trans. Netw. Serv. Manag., vol. 17, no. 2, pp. 876–889, Jun. 2020, doi: 10.1109/TNSM.2020.2971776.

[10] M. Zhu, K. Ye, and C.Z. Xu, “Network anomaly detection and identification based on deep learning methods,” in Cloud Computing – CLOUD 2018, M. Luo and L.J. Zhang, Eds., Cham: Springer, 2018.

[11] S. Alzahrani and L. Hong, “Detection of distributed denial of service (DDoS) attacks using artificial intelligence on cloud,” in 2018 IEEE World Congr. Services (SERVICES), San Francisco, CA, USA, 2018, pp. 35–36.

[12] N. Ahuja, G. Singal, D. Mukhopadhyay, and N. Kumar, “Automated DDoS attack detection in software-defined networking,” J. Netw. Comput. Appl., vol. 187, p. 103108, 2021.

[13] “DDoS 2019 Dataset,” [Online]. Available: https://www.unb.ca/cic/datasets/ddos-2019.html.

[14] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, and D. Siracusa, “Lucid: A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection,” IEEE Trans. Netw. Serv. Manag., vol. 17, no. 2, pp. 876–889, Jun. 2020, doi: 10.1109/TNSM.2020.2971776.

[15] M.A. Setitra, M. Fan, B.L. Agbley, and Z.E. Bensalem, “Optimized MLP-CNN model to enhance detecting DDoS attacks in SDN environment,” Network, vol. 3, no. 4, pp. 538–562, Dec. 2023.