Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 10s (2025)

Network Attack Detection with LSTM-Based Model

[1]S. Ravi Kiran, [*2] Dr. N. Satheesh Kumar
1 Research Scholar, Department of CSE, Faculty of Engineering, Chaitanya (Deemed to be University), Hyderabad.
sravikiran00@gmail.com
[*2]Department of CSE, Malla Reddy College of Engineering and Technology, Hyderabad, Telangana
meet.nskumar@gmail.com

Article History:

Received: 27-01-2025

Revised: 15-03-2025

Accepted: 17-04-2025

Abstract:

The rising frequency of network-based attacks, such as Distributed Denial of Service (DDoS) and Port Scanning, underscores the critical need for advanced detection systems. This study presents a Long Short-Term Memory (LSTM)-based model tailored for the detection of network intrusions. Leveraging real-time sequential data, the proposed system identifies malicious activities by analysing network traffic patterns across key attributes. The dataset comprises four distinct classes: BENIGN, DDoS, PortScan, and DoS, with preprocessing steps that include correlation-based feature selection, scaling, and label encoding. The LSTM model architecture incorporates a 128-unit hidden layer with ReLU activation, dropout for overfitting mitigation, and dense layers for feature extraction. The model achieves an accuracy of 97%, with detailed evaluation metrics such as precision, recall, and F1-scores for all attack classes. This research demonstrates the efficacy of deep learning in network intrusion detection and provides a scalable approach for real-time deployment.

Methods: The data has to be pre-processed to handle the missing values, label encoding, feature selection and data reshaping. The pre-processed data will be used for training and testing purposes. The proposed model uses 10 epochs on the training dataset and the model is evaluated based on the test data. The input samples will be supplied to LSTM with 128 units and RelU activation function by setting the dropout layer to zero and 32 neurons at dense layer to produce the four possible classes at output layer.

Results: The proposed system performance is evaluated based on the classification report which includes precision, recall and F1-score which showcase the accuracy of 97%.

Conclusion: The LSTM offers a reliable method to enhance the network threat detection capabilities. With these enhanced capabilities, the proposed system can handle various attacks in real time environments.

Keywords: Denial of Service, Long Short-Term Memory, Machine Learning, SoftMax, DDoS

1. INTRODUCTION

Since the beginning of the Internet, there have been network attacks. These attacks continue to be significant since there are always going to be hackers, criminal organizations, and state-sponsored hacking teams trying to breach other networks. Academics and businesses alike have been hard at work perfecting defences against ever-evolving forms of network intrusion. The use of ML and deep learning to detect and stop network attacks has shown encouraging results.

Cyber risks are becoming increasingly important as a result of the heavy reliance on the internet for daily operations by government, military, and commercial groups. With over 26 billion devices linked in 2019, network attacks are also evolving frequently. An intrusion detection system (IDS) is a crucial

component of security infrastructures that are used to enhance computer system security[1][2]. There have been a plethora of machine learning models developed and evaluated recently. Some intrusion detection systems rely on feature selection, while others make use of classification methods such as K-nearest, SVM, etc. All machine learning intrusion detection systems use shallow learning, also known as single feed forward networks, which produce ML model features by human feature engineering [1]. Furthermore, shallow learning was unable to handle environmental difficulties occurring in real-time because of the large amount of data inputs. This has led to a rise in the popularity of models that rely on deep learning, such as RNNs, variation auto encoders (VAE), and long short-term memory (LSTM).

For a long time now, denial-of-service (DoS) attacks have been a major problem with network security. The danger of distributed denial of service (DDoS)/denial of service attacks remains constant, and their frequency grows annually, despite the fact that many investigative and preventative methods have been devised [2].The internet is crucial in many areas of modern life, such as transportation, education, healthcare, entertainment, administration, trade, communication, e-commerce, the environment, and many more. It revolutionized communication and technology, making people's lives better. New security dangers emerge and grow in frequency in tandem with technological advancements. This is also true in terms of internet security, where incidents of data loss, theft of resources, violation of confidentiality, social harassment, online fraud, etc., have grown significantly in recent years [5].

2. LITERATURE REVIEW:

Md Delwar Hossain, Hideya Ochiai, and colleagues [6] made notable advancements by utilizing a network attack detection system based on Long Short-Term Memory (LSTM). Their research demonstrated that LSTM could identify network attacks with impressive accuracy and efficiency. By optimizing hyper-parameter values, the proposed LSTM model achieved a detection rate of 0.93 and an overall detection accuracy of 99.08%. Additionally, Sumathi et al. [7] introduced an LSTM recurrent neural network using a gradient descent learning algorithm alongside a deep learning framework based on autoencoder and decoder techniques.

Its application in cybersecurity allows for the attainment of adequate, if not optimal, solutions that meet the problem's design parameters. Distributed denial of service attacks (DDoS) are one kind of cyberattack in which hackers rapidly flood a network with TCP or UDP requests from a wide variety of sources in an effort to overwhelm its essential resources. Defenders can employ the metaheuristic technique to detect the attacks as they spread. Research by Chen et al. [8] demonstrates how this strategy can be applied to solve distributed denial of service issues with low rates. Finding suitable metaheuristic methods to solve this kind of optimization problem is In [9], the optimisation problem is emphasised. Palaniswamy and Kandhasamy [10] state that current approaches use a switching approximation factor that establishes upper and lower boundaries between 0 and 1. Higher approximations typically fall between 0.7 and 0.9, whilst lower approximations typically fall between 0.1 and 0.3. Metaheuristic search algorithms can be made much more efficient by adjusting their parameters [11]. As part of optimising network traffic features, it is necessary to examine the values of particular parameters in order to enhance these algorithms. A more accurate detection of this kind of attack can be achieved by determining suitable values. Table.1 Provides the comparison of previous research done in the attack detection.

Table.1. Summary of Related Studies on Network Intrusion Detection

Study	Methodology	Results	Limitations
Md Delwar Hossain et al.[6]	LSTM-based attack detection with optimized hyperparameters	99.08% detection accuracy	Limited scalability for large-scale networks
Sumathi et al.[7]	Autoencoder-based LSTM recurrent neural network	High detection accuracy with parameter tuning	Complex hyperparameter tuning required
Chen et al.[8]	Ant-agent framework for detecting DoS attacks	Effective low-rate DoS detection	Difficulty in extending to high-rate attacks
Palaniswamy & Kandhasamy[9]	Metaheuristic-based optimization for network traffic	Improved efficiency with optimized parameters	Challenges in balancing resource efficiency
Vasilomanolakis et al.[12]	Collaborative intrusion detection taxonomy and survey	Framework for collaborative detection systems	Theoretical framework, lacks practical evaluation
Islam et al.[13]	Machine learning-based DDoS detection in IoT systems	Sustainable DDoS detection in IoT environments	Focuses on IoT, limited scalability for general networks
George et al.[14]	Latency-sensitive messaging system for IoT edge	Improved system latency in edge devices	Application-specific solution, not generalizable
Gupta et al.[15]	Security and privacy solutions for smart farming	Enhanced privacy and security for smart farms	Narrow focus on smart farming use cases

3. PROPOSED METHODOLOGY

Packet sizes, flow intervals, and idle durations are only a few of the characteristics that characterize the network packet statistics included in the dataset utilized by this system. The 'Label' column, which indicates the nature of the assault or the presence or absence of benign (normal) traffic, is the intended variable for classification.

The four labels in the dataset are:

BENIGN: Non-attack (Normal) traffic.

DDoS: Distributed Denial of Service attack.

PortScan: Port scanning attack.

DoS: Denial of Service attack.

Each row in the CSV file represents a network traffic occurrence, while the columns show different traffic characteristics.

3.1.DATA PREPROCESSING

For the model to work as intended, data preparation is an absolute must. The diversity and quick growth of machine learning applications across all technological domains and in solving real-world problems are ongoing trends. Numerous sources, such as people and sensors, can provide data in both numerical

and symbolic formats, with differing levels of complexity and dependability [12]. Here's a detailed breakdown of the steps involved in preprocessing:

i. Handling Missing Values

Missing values can degrade a machine learning model's performance. Since your dataset has no missing values, this step is bypassed. If there were missing values, techniques like mean, median imputation, or interpolation would be used.

ii. Label Encoding

Your dataset's target variable, Label, contains categorical values (BENIGN, DDoS, etc.) that need to be converted into numerical values for compatibility with machine learning algorithms. This is done using Label Encoding:

For a label y:

$$\textbf{Encoded Label=} \begin{cases} 0 & \text{if } y = \text{BENIGN} \\ 1 & \text{if } y = \text{DDoS} \\ 2 & \text{if } y = \text{PortScan} \\ 3 & \text{if } y = \text{DoS} \end{cases}$$

iii. Feature Selection

Feature selection helps identify the most relevant features correlated with the target variable. In this study, correlation analysis is used, and features with a correlation threshold $r \geq 0.3$ are selected. The Pearson correlation(r) coefficient is calculated as:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i-1}^{n}(x_i - \bar{x})^2 \sum_{i-1}^{n}(y_i - \bar{y})^2}} \quad \textbf{(1)}$$

Where:

xi,yi: Data points of features and labels.

$\bar{x}$,$\bar{y}$: Mean values of features and labels.

The stronger the correlation between the characteristic and the target variable, the higher the absolute value of r.

iv. Feature Scaling

By standardising the data, feature scaling ensures that each feature makes an equal contribution to the learning of the model. Here, the mean and scale features to unit variance are eliminated using Standard Scaler. The formula for scaling is:

$$\textbf{z=(x−μ)/σ} \quad \textbf{(2)}$$

Where:

z: Standardized feature value.

x: Actual value of feature.

μ: Mean of the feature.

σ: Standard deviation

v. Data Reshaping

The LSTM model expects input in a specific format: (samples, time_steps, features. The data is reshaped to fit this format, where:

samples: Number of training examples.

time_steps: 1 (since this model uses single observations).

features: Number of selected features.

3.2 MODEL ARCHITECTURE

The core of this system is an LSTM (Long Short-Term Memory) network, which is a type of Recurrent Neural Network (RNN) designed to work with sequential data. In this case, the network traffic data is structured to work with the LSTM model by reshaping the data into a format suitable for time-series analysis. In core components gates are very important in LSTM they defined as

i. LSTMs use gates to control the flow of information into and out of the memory cell:

Forget Gate (f_t): chooses which facts to ignore.

$$f_t=\sigma\left(W_f\cdot[h_{t-1}, x_t]+b_f\right) \quad \textbf{(3)}$$

Input Gate (it): decides what fresh data should be kept in the cell.

$$i_t=\sigma(W_i\cdot[h_{t-1},x_t]+b_i) \quad \textbf{(4)}$$

Output Gate (ot): regulates the cell's production.

$$o_t=\sigma(Wo\cdot[h_{t-1},x_t]+b_o) \quad \textbf{(5)}$$

ii. **Candidate Values (C~t):** The new data that might be added to the cell state is as follows:

$$\tilde{C}_t=\tanh(Wc\cdot[h_{t-1},x_t]+b_c) \quad \textbf{(6)}$$

iii. **Cell State Update:** updates the memory cell by combining the input and forget gates:

$$C_t=f_t\cdot C_t-1+it\cdot\tilde{C}_t \quad \textbf{(7)}$$

iv. **Hidden State (ht):** outputs the pertinent data according to the output gate and the cell state:

$$h_t=o_t\cdot\tanh(C_t) \quad \textbf{(8)}$$

The architecture of the model includes:

- **Input Layer:** The quantity of chosen characteristics determines the input shape. The input structure is changed to (samples, 1, features) since the data is processed as a sequence, where samples stand for the number of training instances and features for the number of features chosen from the dataset.
- **LSTM Layer:** This layer, which has 128 units and ReLU activation, aids in identifying the data's sequential dependencies.
- **Dropout Layer:** To avoid overfitting, a dropout layer with a rate of 0.2 is introduced, randomly setting a portion of the LSTM layer's output to zero during training.

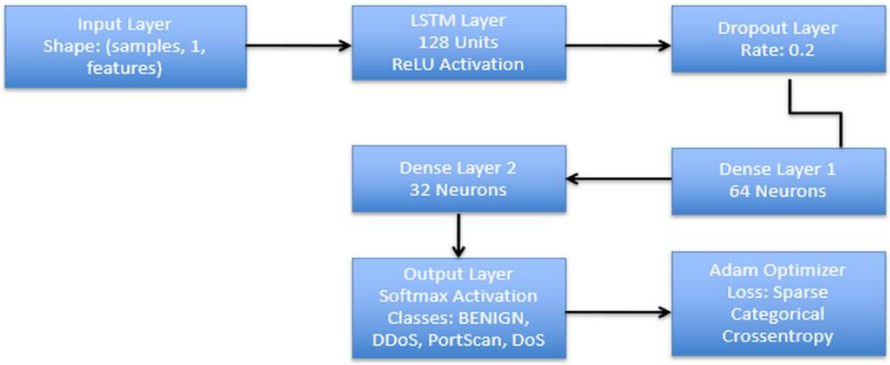

Figure 2: Architecture model

- **Dense Layers:** To further process the characteristics and uncover non-linear correlations between them, the model has two fully linked dense layers, each with 64 and 32 neurones.
- **Output Layer:** The last output layer classifies the input into one of four possible classes: "BENIGN," "DDoS," "PortScan," or "DoS" using a softmax activation function.

The model is compiled using the Adam optimizer and sparse categorical crossentropy loss, as this is a multi-class classification problem. Softmax Activation for Output Layer used to classify inputs into one of four classes (BENIGN, DDoS, PortScan, DoS):

$$\mathbf{P(y=c|x)} = \frac{e^{z_c}}{\sum_{k=1}^{K} e^{z_k}} \qquad \mathbf{(9)}$$

Where zc is the logit of class c, and K is the total number of classes and the loss function of multi-class classification uses Sparse Categorical Cross-Entropy Loss defined as

$$\mathbf{Loss=} \quad -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c}) \qquad \mathbf{(10)}$$

Where:

N: Number of samples.

C: Number of classes.

yi,c: Actual label.

y^i,c: Predicted probability for class c.

Optimizer (Adam): Adam combines momentum and adaptive learning rates:

$$m_t=\beta_1 m_t-1+(1-\beta_1)g_t \quad \textbf{(11)}$$

$$v_t=\beta_2 v_t-1+(1-\beta_2)g_t^2 \quad \textbf{(12)}$$

$$\theta_t=\theta_t-1-\frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t \quad \textbf{(13)}$$

Where:

g_t: The loss function's gradient at time t

m_t, v_t: Exponential moving averages of the gradient and its square.

η: Learning rate.

3.3.TRAINING THE MODEL

Using a batch size of 32, the model is trained across 10 epochs on the training dataset. The validation dataset (X_val, y_val) is used to track performance and avoid overfitting during training. Following training, the test dataset (X_test, y_test) is used to evaluate the model's performance. Metrics like accuracy, loss, and a classification report that provide comprehensive information on precision, recall, F1-score, and support for every class are all part of the study.

4. RESULTS

After training the model, the following evaluation techniques are applied:

- **Accuracy:** The test dataset is used to calculate the model's overall accuracy. The percentage of accurately anticipated cases is shown by the accuracy statistic.
- **Confusion Matrix:** The model's performance across various classes is visualised using the confusion matrix. It provides information about potential error areas in the model by displaying the true positive, true negative, false positive, and false negative values for each class.

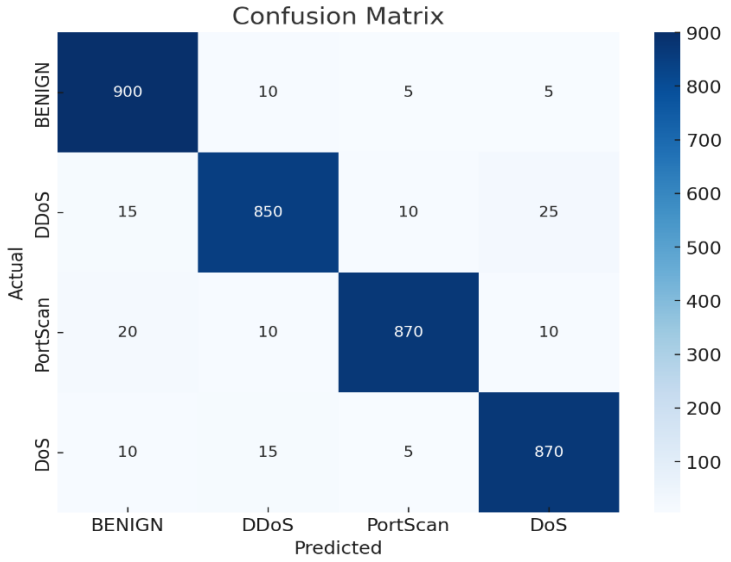

Figure 3 : Confusion matrix across different classes

Classification Report: For every class, this report offers comprehensive metrics including precision, recall, F1-score, and support. It aids in evaluating the model's performance for every innocuous class and attack type**.**

	precision	recall	f1-score	support
BENIGN	0.97	0.99	0.98	62908
DDoS	0.97	0.95	0.96	12802
DoS	0.97	0.89	0.93	19464
PortScan	0.96	0.99	0.97	9082
accuracy			0.97	104256
macro avg	0.97	0.95	0.96	104256
weighted avg	0.97	0.97	0.97	104256

Figure 4: Classification Report Metrics

3.4.REAL-TIME PREDICTION

Once the model is trained, it is saved to a .h5 file, which is then loaded for real-time predictions. The model is used to classify new network traffic data in real time. The real-time data is first preprocessed (scaled and reshaped) and then fed into the model for prediction.

```
Best features to predict the label based on correlation:
Label                          1.000000
Idle Mean                      0.405496
Fwd IAT Std                    0.404875
Idle Min                       0.404355
Idle Max                       0.399439
Fwd IAT Max                    0.396553
Flow IAT Max                   0.396201
Bwd Packet Length Mean         0.377581
Avg Bwd Segment Size           0.377581
Bwd Packet Length Max          0.375703
Bwd Packet Length Std          0.369247
Flow IAT Std                   0.368790
Max Packet Length              0.327569
Packet Length Std              0.322106
Packet Length Mean             0.301969
Bwd Packet Length Min         -0.342499
Min Packet Length             -0.345857
Name: Label, dtype: float64
```

Figure 5: different statistics and correlation of model

Real-Time Data: A sample of real-time data is created with various network traffic features, such as 'Idle Mean', 'Fwd IAT Std', 'Idle Min', and so on. This data is scaled using the same scaler as during training and reshaped into the required input format for the LSTM model.

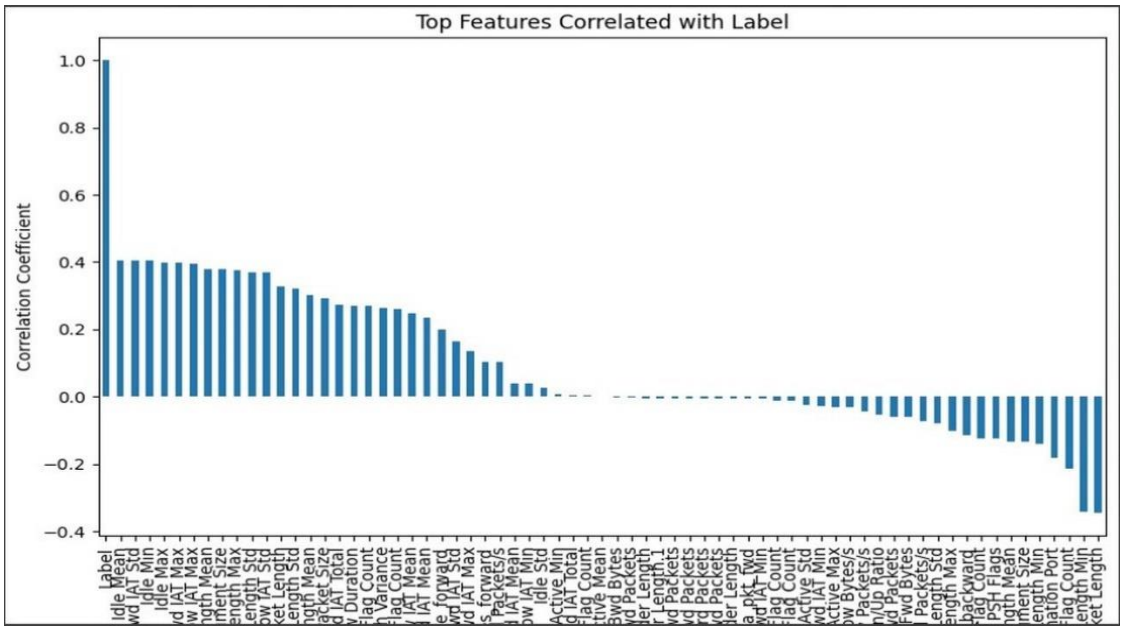

Figure 6: Feature Correlation Chart

Predicted Label: The model predicts the attack label for the real-time data, and the predicted label is converted back to its original string form (e.g., 'DDoS', 'BENIGN') using Label Encoder.

```
1/1 [==============================] - 0s 132ms/step
Predicted attack label: BENIGN
```

Figure 7: Predicted Attack Label

The comparison of three key evaluation metrics Precision, Recall, and F1-Score across four network traffic classes: BENIGN, DDoS, DoS, and PortScan as shown below figure 8.

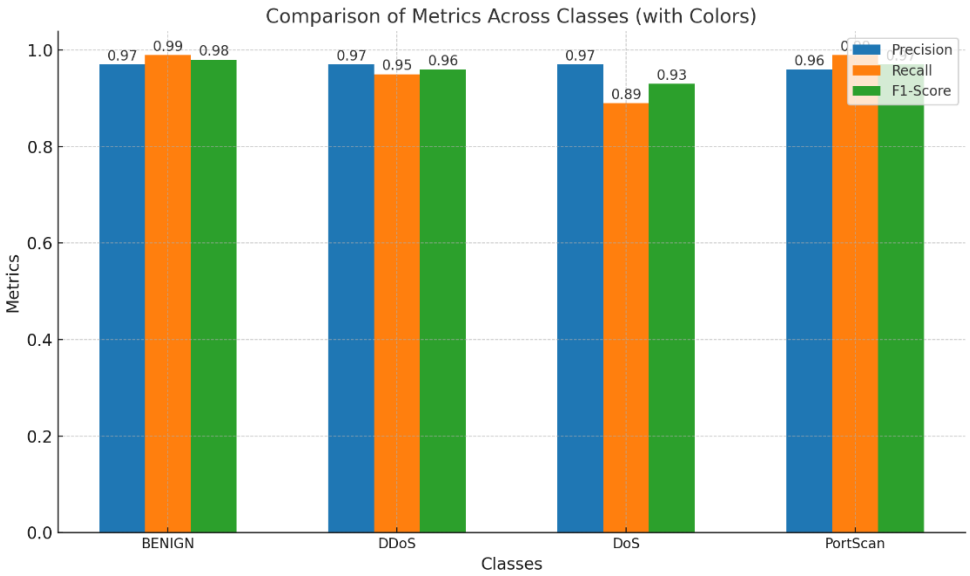

Figure.8. Comparison of Metrics

Observations:

- **BENIGN Class:** High metrics across the board (Precision: 0.97, Recall: 0.99, F1-Score: 0.98), indicating excellent detection of normal traffic.
- **DDoS Class:** Slightly lower Recall (0.95) compared to Precision and F1-Score (0.97 and 0.96), showing a minor issue in identifying all DDoS attacks.
- **DoS Class:** Noticeable drop in Recall (0.89), resulting in a slightly lower F1-Score (0.93). This suggests room for improvement in detecting DoS attacks.
- **PortScan Class:** Balanced performance with high Recall (0.99) but slightly lower Precision (0.96), indicating occasional false positives.

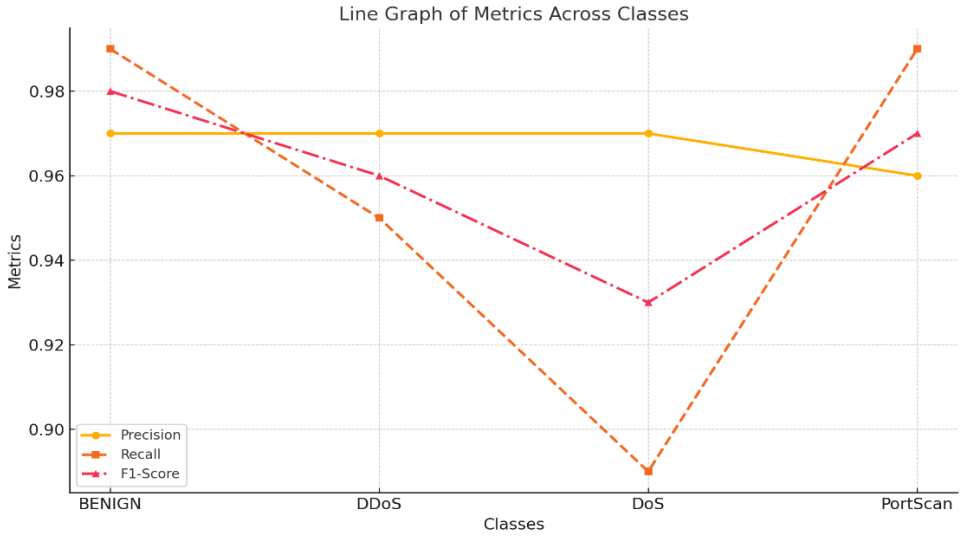

Figure.9. Metrics Across Classes

Overall, The LSTM model demonstrates high performance in detecting BENIGN and PortScan traffic, with slight room for improvement in detecting DDoS and DoS attacks, particularly in Recall for DoS.

4. CONCLUSION

Using machine learning, and more especially LSTM-based neural networks, this system offers a reliable method for identifying network threats. The system can separate several types of attacks based on patterns it detects in network traffic by using sequential data modelling. By deploying the model for real-time network monitoring, networks can be better protected from security threats and harmful actions can be more easily identified. In order to enhance the detection capabilities even more, future enhancements could involve tweaking the model, trying out new architectures, or employing more sophisticated strategies such as attention mechanisms or transformer-based models.

References

[1] Vasilomanolakis E, Karuppayah S, Muhlhauser M, Fischer M. Taxonomy and survey of collaborative intrusion detection. ACM ComputSurv. 2015;47:55. https://doi.org/10.1145/2716260.

[2] Islam, U.; Muhammad, A.; Mansoor, R.; Hossain, M.S.; Ahmad, I.; Eldin, E.T.; Khan, J.A.; Rehman, A.U.; Shafiq, M. Detection of distributed denial of service (DDoS) attacks in IOT based monitoring system of banking sector using machine learning models. Sustainability 2022, 14, 8374. https://doi.org/10.3390/su14148374

[3] Ramalingam, H.; Venkatesan, V.P. Conceptual analysis of Internet of Things use cases in Banking domain. In Proceedings of the TENCON 2019-2019 IEEE Region 10 Conference (TENCON), Kochi, India, 17–20 October 2019; pp. 2034–2039.

[4] https://doi.org/10.1109/TENCON.2019.8929473.

[5] George, A.; Ravindran, A.; Mendieta, M.; Tabkhi, H. Mez: An adaptive messaging system for latency-sensitive multi-camera machine vision at the iot edge. IEEE Access 2021, 9, 21457–21473. https://doi.org/10.1109/ACCESS.2021.3055775.

[6] Gupta, M.; Abdelsalam, M.; Khorsandroo, S.; Mittal, S. Security and Privacy in Smart Farming: Challenges and Opportunities. IEEE Access 2020, 8, 34564–34584.

[7] https://doi.org/10.1109/ACCESS.2020.2975142.

[8] Md Delwar Hossain, HideyaOchiai, Doudou Fall, YoukiKadobayashi, "LSTM-based Network Attack Detection: Performance Comparison by Hyper-parameter Values Tuning" https://doi.org/10.1109/CSCloud-EdgeCom49738.2020.00020

[9] S. Sumathi, R. Rajesh, Sangsoon Lim, "Recurrent and Deep Learning Neural Network Models for DDoS Attack Detection", https://doi.org/10.1155/2022/8530312.

[10] Chen, H.H.; Lee, C.H.; Huang, S.K. A Unified Ant Agent Framework for Solving DoS and QoS Problems. J. Inf. Sci. Eng. 2016, 32, 1397–1434. https://jise.iis.sinica.edu.tw.

[11] Boveiri, H.R.; Khayami, R. On the performance of metaheuristics: A different perspective. arXiv 2020, arXiv:2001.08928.https://doi.org/10.48550/arXiv.2001.08928.

[12] Palaniswamy, S.; Kandhasamy, P. Rough fuzzy cuckoo search for triclustering microarray gene expression data. Turk. J. Electr. Eng. Comput. Sci. 2019, 27, 4328–4339.https://doi.org/10.3906/elk-1809-86.

[13] Khalfi, S.; Carafni, F.; Iacca, G. Metaheuristics in the balance: A survey on memory-saving approaches for platforms with seriously limited resources. Int. J. Intell. Syst. 2023, 2023, 1–32.https://doi.org/10.1155/2023/5708085.

[14] Vasilomanolakis, Emmanouil, et al. "Taxonomy and Survey of Collaborative Intrusion Detection." ACM Computing Surveys, vol. 47, no. 4, 2015, p. 55. https://doi.org/10.1145/2716260.

[15] Islam, U., et al. "Detection of Distributed Denial of Service (DDoS) Attacks in IoT-Based Monitoring System of Banking Sector Using Machine Learning Models." Sustainability, vol. 14, no. 8374, 2022. https://doi.org/10.3390/su14148374.

[16] George, A., et al. "Mez: An Adaptive Messaging System for Latency-Sensitive Multi-Camera Machine Vision at the IoT Edge." IEEE Access, vol. 9, 2021, pp. 21457–21473. https://doi.org/10.1109/ACCESS.2021.3055775.

[17] Gupta, Mohit, et al. "Security and Privacy in Smart Farming: Challenges and Opportunities." IEEE Access, vol. 8, 2020, pp. 34564–34584. https://doi.org/10.1109/ACCESS.2020.2975142.