

Hybrid Deep Learning Framework For Robust Network Traffic Classification

^{*1}Varinder Kaur and ²Amandeep Kaur Virk

Sri Guru Granth Sahib World University, Fatehgarh Sahib, Punjab, India

^{*}Corresponding Author Varinder Kaur

Co-author Amandeep Kaur Virk

Abstract

As network traffic is growing at a fast rate, it is important to predict future traffic patterns in order to manage and optimize the network. This study provides a hybrid deep learning architecture, which is a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks and which can improve the accuracy of network traffic prediction. The feature extraction uses CNN that captures spatial dependencies in the data and the temporal dependencies are captured by using LSTM that allows the framework to successfully learn long-term trends in network traffic. The proposed hybrid model is better than the classical deep learning model, which relies on the ability of CNN to obtain the required characteristics and the power of LSTM to analyze sequential dependencies. The framework is far superior to the current deep learning models in that it has solved the problem of complexity of feature extraction and also sequence learning, which makes it a formidable tool in the real-time network traffic prediction. The KDD dataset of network traffic prediction is used to implement the proposed model. The proposed model is tested regarding accuracy, precision, and recall. The findings prove that the hybrid CNNLSTM model with an accuracy of 98% are accompanied by a precision of 97% and a recall of 96% meaning that the model has a high capacity to detect normal and attack traffic correctly. The proposed method also presents a significant improvement over the baseline deep learning models in the detection of minority attack classes, which, in turn, confirms its practical use in the real world.

Keywords: network traffic prediction; KDD; CNN; bidirectional LSTM.

1. Introduction

The fast development of the internet makes network applications popular in most industries. As the internet technology continues to improve and it rapidly adopts the latest innovations, it is an added boost to the development and it provides a push to most industries. Nonetheless, internet technology has its dark and bright sides as is the case with other technological advancements. It has the capacity to increase the growth of the global economy, which is one of its advantages [1,2]. These attacks are dangerous to the national information security in extreme cases. In the process of information exchange and interaction on the cyberspace, the network traffic is a crucial conduit [3,4]. This leads to the network traffic storing a great deal of important information. As the amount of network data is huge, there is a wide range of useful techniques and methodologies that can be used to analyze such traffic. Through these approaches, one is able to know the present situation in the evolution of the network, identify anomalous occurrences in time and precise and do the right action to keep the system secured. Network traffic classification is a procedure of determining certain applications or activities based on their pattern of network traffic. This is a task that is significant to good network management and security [5,6]. The first attempt at network traffic classification is based on utilization of protocols of the transport layer, e.g. UDP or TCP port numbers. It is an easy implementation method with a low cost of computation, which has been effective in the identification of applications running on particular ports. But, as more applications and protocols are implemented, and as port-hopping and obfuscation mechanisms are being more widely used, the effectiveness of traffic classification by

port-identification alone is reduced [7,8]. As a result, such an approach is no longer viewed as reliable and can only be discussed as an auxiliary instrument of the network traffic classification. The idea of traffic classification based on the statistical properties of the network traffic and machine learning algorithms attracts a lot of attention among researchers, particularly with the development of science and technology as well as enhancement in the processing power in computers [9,10].

The most popular machine learning algorithms that are used in network traffic classification are Bayes, SVM, Decision Tree and KNN. These algorithms can be used to classify network traffic by training models on known categories meaning that the models can be used to recognize and classify new traffic within the network. Classification methods based on machine learning usually demand a lot of storage and computation power, so they are not easily applicable to resource-constrained systems such as vehicles, home gateway and mobile phones [11,12]. Real time, precise network traffic classifier is also valued to network management and intrusion detection system, which creates an urgent demand of novel classification techniques. The DL approaches avoid feature selection that requires labor and the personal accessibility of features by extracting and selecting features when training. With such benefits, DL-based techniques successfully learn the correlation between raw traffic data and the target output in an end-to-end fashion without necessarily having to involve a huge amount of manual labor or confidential knowledge [13,14]. CNNs are popular deep learning systems which find features of the input data by means of convolutional operation layers. The characteristics that are gotten at the deeper layers of a network, on the other hand, are more abstract than the characteristics retrieved at the shallower layers [15,16]. It should be noted, however, that it is not always possible to interpret the features that are derived on the middle layers of a network and see meaningful concepts as with tasks such as face recognition. 1D-CNNs are the best option in cases of network traffic classification. This is attributed to the fact that 1D-CNNs learn spatial dependencies between neighboring Bytes in network packets and, consequently, identify discriminative patterns that differing protocol/application classes follow and, thus, correct traffic classification.

2. Literature Review

M. Abbasi et al. (2025) introduced the AWEE (Adaptive Weight Ensemble-of-Ensemble Learning) technique as a novel solution to the issues of idea drift and class imbalance in dynamic data settings [17]. This approach demonstrated the combined power of numerous modelling methodologies by integrating several ensemble layers with an adaptive weight adjustment mechanism. The model used a sliding window-based validation procedure to increase its flexibility and resilience, thereby resolving the problem of concept drift in constantly shifting data streams. AWEE showed remarkable performance through extended testing on benchmark datasets, attaining accuracy rates higher than 98%. These outcomes demonstrated how well it handled class imbalance issues under a variety of network traffic scenarios.

X. Wang et al. (2025) developed RoNeTC to address this issue more reliably and accurately [18]. By enhancing feature extraction and using uncertainty estimation to evaluate the dependability of classification results, this approach produced accurate classification. To improve accuracy, they divided each flow packet into three distinct viewpoints for feature extraction, incorporating local and global feature representations from several packets. Additionally, they ensured reliable and resilient results by dynamically integrating classification outcomes from many perspectives and analysing

decision uncertainty to classify both known and unknown flows. With an average improvement of 25.94% in F1 score, RoNeTC outperformed four top state-of-the-art algorithms in six open-set scenarios, demonstrating its superior efficacy in open-set network traffic classification.

Z. Wang et al. (2024) developed a TC (Traffic Classification) framework for smart home networks that uses FL (Federated Learning) to protect the privacy of traffic data through local TC model inference and training. They first created a traffic labelling technique based on DPI (Deep Packet Inspection) on edge residential gateways, which served as FL nodes. Alongside benchmark techniques, the suggested approach was evaluated on real-world and public datasets. Experimental results showed that the approach preserved data privacy and improved the model's reliability while achieving strong performance with a small number of samples.

W. Lin et al. (2024) introduced a novel technique to address these issues by efficiently extracting important features from unprocessed traffic data while eliminating unnecessary information: the IBNN (Information Bottleneck Neural Network) [19]. The IBNN architecture enhanced classification accuracy and robustness, particularly when handling encrypted communication, by striking a compromise between model complexity and the retention of significant features. On the ISCX 2012 and UNB CIS datasets, the suggested approach outperformed current methods by 5-7% in accuracy and F1 score by concentrating on information-driven feature extraction and compressing pertinent data. This led to increased adaptation and durability in a variety of network situations. IBNN's efficacy was confirmed by numerous trials, making it a major advancement in NTC (network traffic classification).

G. Hu et al. (2023) presented TCGNN, a new Traffic Classification model based on GNN (Graph Neural Networks) that made use of the idea of examining network packets from a graph perspective. Each network packet was first transformed into an undirected graph by TCGNN [20]. Lastly, TCGNN identified unfamiliar network packets with a high degree of accuracy by utilising the GNN's remarkable graph representation learning capabilities. TCGNN considerably surpassed conventional packet-level traffic classification techniques in terms of efficacy, according to extensive trials carried out on two real-world traffic classification datasets.

Y. Wang et al. (2022) presented Ulfar, a multi-scale feature attention technique for classifying network traffic that used CNNs (convolutional neural networks) as the central part of the deep packet analysis model [21]. Ulfar classified traffic using only one packet per flow. By building high-level, multi-scale n-gram features from raw byte sequences, this neural network model was able to automatically extract these format-related bytes. They also sought to identify the patterns and characteristics that the neural network picked up from the network traffic during the representation learning stage. Two publicly accessible datasets were used to assess Ulfar, and the experimental findings showed that it could accurately classify network traffic.

3. Research Methodology

On the basis of deep learning, a framework is created that predicts network traffic. The framework of the constructed model is supposed to be more precise at predicting the network traffic than the existing deep learning models. The steps of the model proposed can be explained as follows:

Dataset Input and Pre-Processing

The data are gathered in the form of a dataset using Kaggle and pre-processing step is implemented to eliminate any missing or unacceptable values of the data set. NSL-KDD data set has been cleaned, yet it needs certain preliminary feature engineering prior to being fed in the model. Precisely, the categorical characteristics have to be converted into numerical representation so that they can be conceptualized as a vector in Euclidean space: We encode the three categorical attributes ('protocol type,' 'service,' and 'flag') using a LabelCount encoder and order the categories in terms of the frequency of occurrence within the feature. The advantages of labelCount are that it is not sensitive to outliers and on instances where some of the attributes have many categories, it shrinks dimensions. The next thing after assigning a numerical value to each categorical characteristic is normalization. The reason is that features which have been recorded on different scales do not contribute the same to the analysis and are likely to be biased in models. Therefore, the standardization converts the data to similar scales (around the middle 0 with a standard deviation of 1) as shown in Equation (1).

$$Z = \frac{x - \mu}{\sigma} \quad (1)$$

Where x is used to denote the individual values within the feature, Z is used to denote the standardized feature and μ and σ are the average and standard deviation of all values, respectively. In the case of the binary classification (normal vs. others) in this dataset the samples in each category are approximately equal. But due to each of the class ratio being approximately 920: 220: 20: 1, the data of incursion in the 4-class is imbalanced seriously. In their pattern recognition and further classification tasks, most machine learning algorithms assume or expect a balanced distribution of classes. In the event that such kind of data skewness takes place, such algorithms cannot provide a true representation of the distributive nature of the data, and therefore the results derived will be inaccurate across classes of data. In this research, SMOTE, which is a complex oversampling method, is used to identify samples to utilize to form new synthetic samples as the framework of the SMOTE class to counterbalance the debilitating impacts of unequal data gathering.

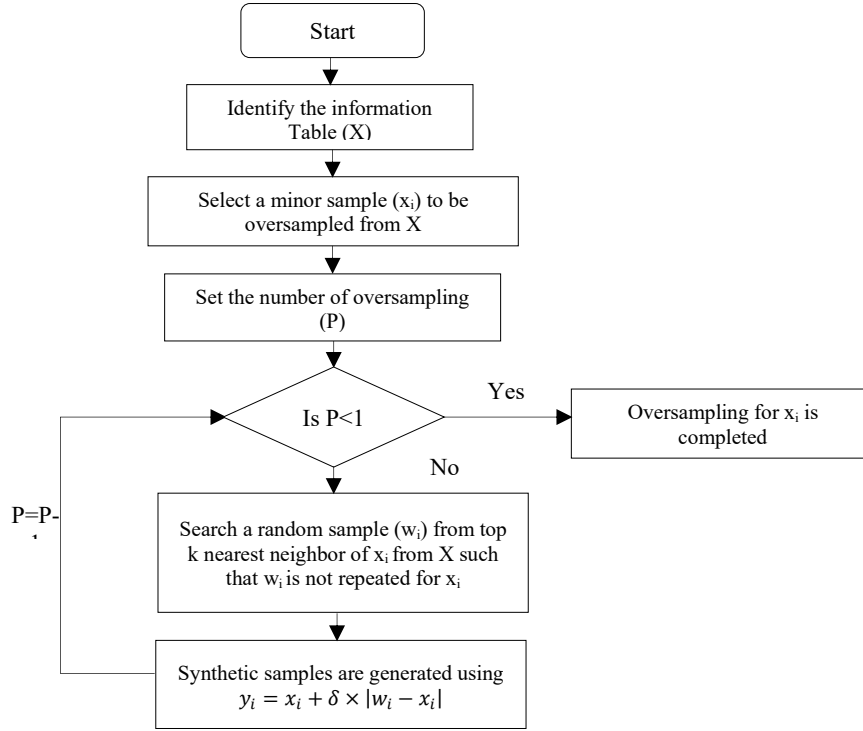


Figure 1. SMOTE Algorithm

Feature Reduction

In this step, Principal Component Analysis (PCA) is applied to reduce the dimensionality of the dataset by minimizing the number of attributes while preserving as much useful information as possible. PCA is a statistical technique that transforms a set of correlated variables into a set of linearly independent variables. This transformation is achieved through an orthogonal linear mapping, which reprojects the original data into a new coordinate system. The principal components are ordered such that the first component captures the maximum possible variance in the data, while the second component captures the next highest variance and is orthogonal to the first, and so on. In this way, PCA ensures that the extracted components are uncorrelated and independent of each other. Formally, $X = \{x_1, x_2, \dots, x_i\}$, where $x_i \in \mathbb{R}^d$ denotes the original data vectors. The goal is to project x_i into a lower-dimensional space $z \in \mathbb{R}^r$ with $r < d$ such that the variance in the projected space is maximized. This is achieved by learning a set of p -dimensional weight vectors $W = \{w_1, w_2, \dots, w_p\}$, with $w_p \in \mathbb{R}^k$, which define the projection directions. The projection of each data vector x_i onto the k -th principal component is given by:

$$t_{k(i)} = W_{(i)} T_{x_i} \quad (2)$$

Here, $t_k(i)$ is a scalar that measures the degree of alignment (or similarity) between the weight vector w_k and the input vector x_i . In cases where W is a weight matrix, the notation w_k may represent a specific row or column corresponding to the k -th projection direction. After observing the condition primary with weight W_1 , then the variance is increased as in Equation (3).

$$W_i = \arg \max_{|w|} = \{\sum_i (x_i \cdot W)^2\} \quad (3)$$

This is an enlarged condition as illustrated in the Equation (4).

$$W_i = \arg \max_{\|w\|=1} \{\|X \cdot W\|^2\} = \arg \max_{\|w\|=1} \{W^T X^T X W\} \quad (4)$$

The symmetric grid which is the $X^T X$ is analyzed successfully after the extraction of the largest eigenvalue of the matrix with W representing the eigenvector. The first principal component is determined by predetermined vectors W_1 which project the original data matrix X onto W_1 in the transformed space. The resulting components are narrowed down to give more segments in the same direction.

Classification

This phase performs an ultimate hybrid deep learning system that incorporates CNN and Bi-LSTM models. CNN is a deep network that mimics and explains the stimuli in a similar manner that the visual cortex of the human brain can explain. It is also a deep neural network comprising of several hidden layers. The neural network is normally used in multiclass classification on the CNN output layer. CNN uses a feature extractor at the training phase instead of extracting the features manually. CNN has a feature extractor that has convolutional and pooling layers the weights of which are trained. The neural network feature extraction in CNN is more effective as the depth of the feature extraction (i. e. the number of layers) increases, yet the learning process becomes more complex. The CNN component performs the task of extracting spatial and local correlation patterns of the input features whereas, by the Bi-LSTM component, it extracts temporal dependencies in both forward and backward directions. Such synergy will not only guarantee that the model not only finds discriminative trends in the data, but also comprehends the sequential relationships that are paramount to forecast network traffic. As contrasted to the conventional LSTM, which has the limitation of only being able to model past dependencies, the Bi-LSTM structure uses future contextual data, which makes it more appropriate to model the bidirectional nature of network events. The feature representation obtained at the outflow of the CNN layers are inputted into the Bi-LSTM which then refines them by attempting to model the dynamics of the traffic flows.

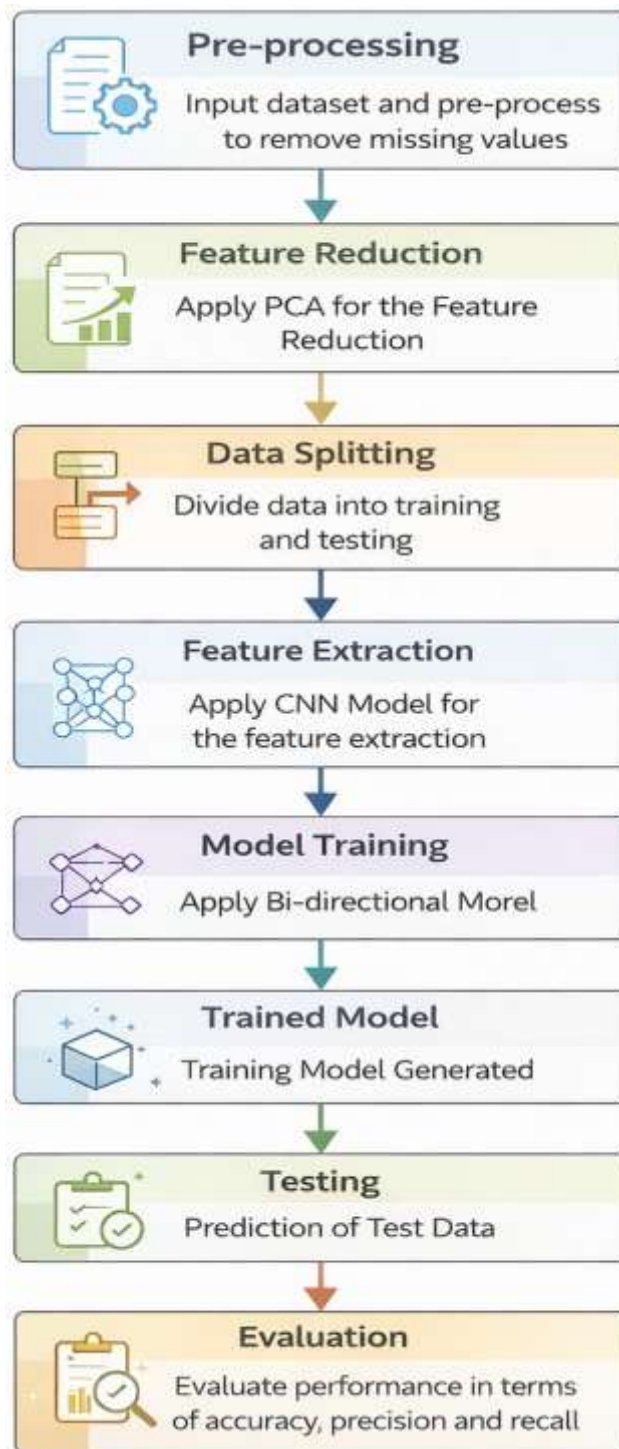


Figure 2. Proposed Model

Figure 2 shows how the proposed model is used to predict the network traffic. The data in the network traffic are first entered into the proposed model and processed to remove the value gaps. Subsequent transformation is the application of PCA algorithm to minimise the features. The data is then separated into two parts namely training and testing. The CNN model is employed to extract the features in the training set and then

the Bi-directional model is employed. Lastly, the performance is measured using a lot of metrics such as accuracy, precision and recall. This leads to the results being achieved. The proposed model also has a pseudo code as illustrated below:

Pseudo Code of Proposed Model
<pre> BEGIN # Step 1: Load and Preprocess Data LOAD dataset PREPROCESS dataset: FOR each instance IN dataset: RESIZE data to fixed dimensions NORMALIZE pixel values SPLIT dataset into training, validation, and test sets # Step 2: Define CNN for Feature Extraction DEFINE CNN_Model: INPUT data FOR each convolutional layer: APPLY convolution operation APPLY activation function (ReLU) APPLY pooling operation FLATTEN output to 1D feature vector # Step 3: Define Bi-LSTM for Sequence Learning DEFINE Bi_LSTM_Model: INPUT extracted features from CNN FORWARD_LSTM processes input sequence BACKWARD_LSTM processes input sequence in reverse CONCATENATE forward and backward outputs APPLY Softmax Regression for classification # Step 4: Train the Hybrid Model COMPILE model with categorical cross-entropy loss function SELECT optimizer (Adam) TRAIN model using training data FOR each epoch: FEED forward training samples CALCULATE loss BACKPROPAGATE error UPDATE weights # Step 5: Evaluate Model Performance EVALUATE model on test dataset CALCULATE accuracy, precision, recall, and F1-score # Step 6: Deploy Model for Classification FUNCTION classify_new_data: PREPROCESS data EXTRACT features using CNN_Model PREDICT class using Bi_LSTM_Model </pre>

RETURN predicted class
END

4. Result and Discussion

The network traffic classification is carried out using the proposed model that is the mixture of CNN and Bidirectional LSTM. KDD data is applied and results are evaluated with respect to accuracy, precision and recall.

Dataset Description

In the study, the KDD dataset is applied in analysing the results. KDD Cup 1999 dataset is a standard benchmark that is usually used to test intrusion detection systems and network security algorithms. It is designed to run in the competition of the KDD Cup and based on the DARPA Intrusion Detection Evaluation Program of 1998 that included simulated network traffic and attack conditions. Every dataset entry is associated with a network connection and has 41 features with the labeling being normal or an attack. These characteristics include rudimentary characteristics such as protocol type and service, content-based characteristics such as the number of attempts to log in and access files and traffic-based measures that are computed during particular time periods. The dataset attacks are categorized into four- DoS, R2L, U2R and Probing. Even though the dataset has been instrumental in the development and testing of numerous machine learning models in the detection of anomalies and misuse, it also has its flaws such as redundancy of data and imbalance in the classes. Table 1 presents the distribution of the data.

Table 1. Dataset Distribution

Class	Instances (Total)	Training Samples	Testing Samples	Percentage (%)
Normal	972,781	50%	50%	~19%
DoS	3,883,370	50%	50%	~79%
Probe	41,102	50%	50%	~0.8%
R2L	1,126	50%	50%	~0.02%
U2R	52	50%	50%	~0.001%
Total	4,898,431	~2.45M	~2.45M	100%

Experimental Setup and Parameter

The model is code executed in Python with TensorFlow/Keras backend. The tests are performed on a workstation computer using Intel i7 processor, 32GBs of memory and the NVIDIA RTX 3080 graphics card. The mean duration of the training process per epoch is about 3.2 minutes; the overall duration of 50 epochs is about 160~ minutes. The model parameters are indicated in table 2.

Table 2. Model Parameters

Parameter	Value
Learning Rate	0.001
Optimizer	Adam
Batch Size	64
Epochs	50
Dropout	0.5
Activation	ReLU (CNN), Tanh (Bi-LSTM)
Loss Function	Categorical Cross-Entropy

Performance Analysis Parameter

The model is proposed using the results of the procedure that is compared against the accuracy, precision and recall. The parameters are described in details as follows:

Accuracy: The percentage of the number of points correctly rated to the total number of points multiplied by 100 is referred to as accuracy and is depicted in Equation (5).

$$\text{Accuracy} = \frac{\text{Number of points correctly classified}}{\text{Total Number of points}} * 100 \quad (5)$$

Precision: In pattern recognition, information retrieval and binary classification, the proportion of the relevant cases in the retrieved cases is referred to as precision (also known as positive predictive value), as defined in Equation (6).

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \quad (6)$$

Recall: Recall is the proportion of the relevant cases that have been recalled out of the total number of relevant cases as indicated in Equation (7).

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \quad (7)$$

Results

The KDD Cup 1999 dataset has 41 attributes as per each network connection which are supposed to be used to distinguish between normal behavior and a large number of possible network intrusions. These characteristics are grouped into three primary categories, which are basic features, content features and traffic features. Basic features are the general properties of the network connection such as the duration, protocol type (e.g. TCP, UDP, ICMP), service type (e.g. HTTP, FTP) and the status flags indicating the success or failure of the connection. They also quantify the amount of data as such fields as source bytes (src_bytes) and destination bytes (dst_bytes). Content features: The payload of the connection is analyzed to determine malicious activities and it contains such things as the number of unsuccessful logins, user is a root user (root_shell), and whether files are created or shells accessed within the session.

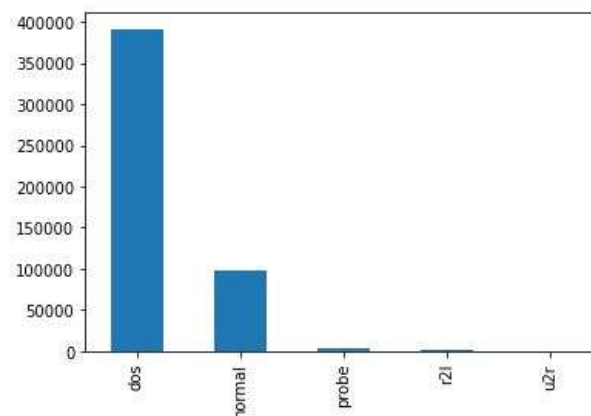


Figure 3. Image List with the Same Label

The implementation is done with the help of KDD dataset as Figure 3 indicates, and this dataset also has five different classes: dos, normal, probe, r2l and u2l. The notable cases are those of normal traffic and rest other classes are counter-balanced.

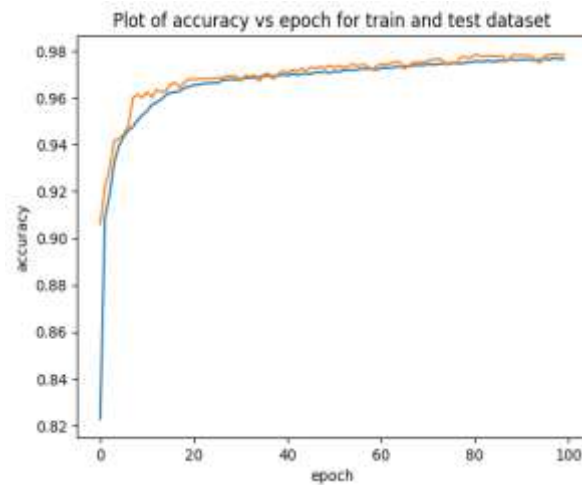


Figure 4. Accuracy of Training and Testing

The model training and test accuracy plotted as in Figure 4 represents varying numbers of values in the epoch. The model training accuracy is realized up to 90% that indicates that the model is highly trained to produce desired results to make the predictions.

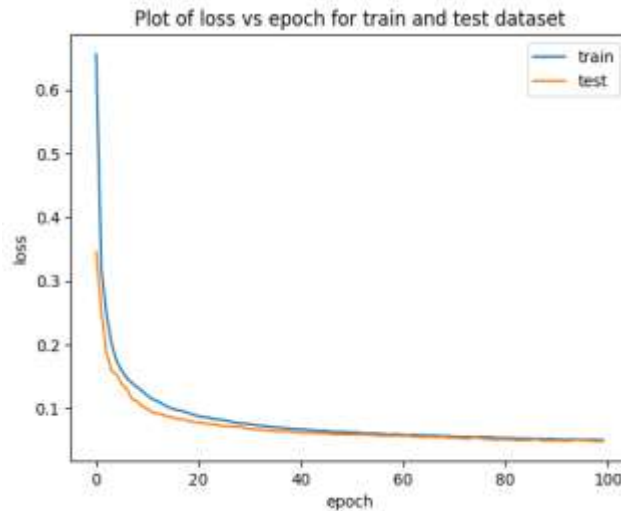


Figure 5. Image List with the Same Label

The number of epoch values is used as a representation of the model training and test loss as shown in Figure 5. Model training has a model loss that is minimized to 2%. The classification report is demonstrated in Table 3.

Table 3. Classification Report

Class Type	Precision	Recall
0	0.66	0.92
1	0.90	0.96

2	0.88	0.66
3	1	1
4	0.95	0.91

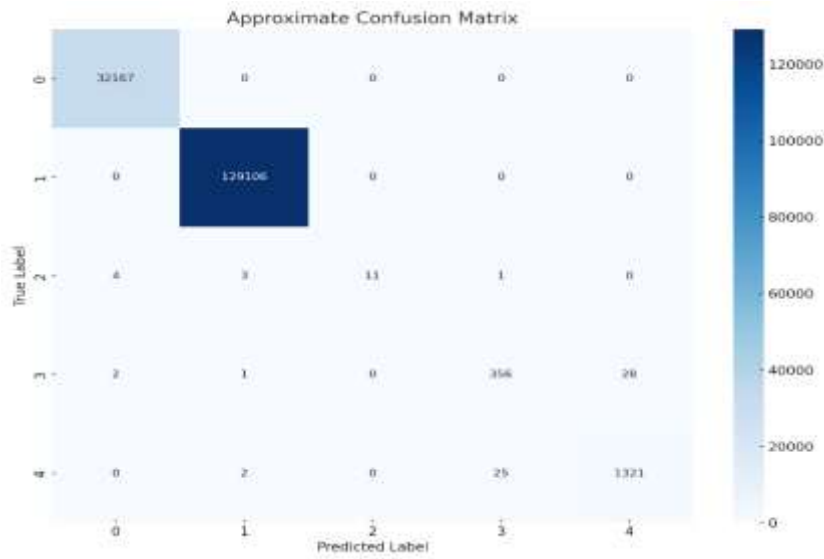


Figure 6. Confusion Matrix

As indicated in Figure 6, the confusion matrix between the actual and predicted labels is drawn. True positive and false positive values of predicted labels are drawn against true labels.

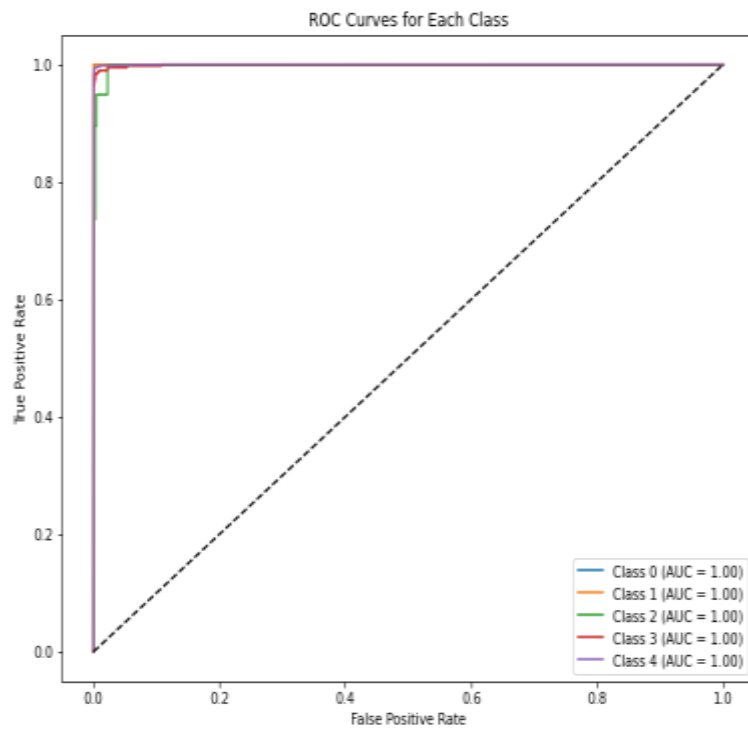


Figure 7. ROC Curve

Figure 7 illustrates the drawing of the AUC-ROC curve by each class of the dataset. False positive rate is displayed in the x axis and true positive rate in the y axis. The ROC curve of every class converge more than 95% of the network traffic prediction. Table 4 indicates performance analysis.

Table 4. Performance Analysis

Models	Accuracy	Precision	Recall	F1-Score
A. Ariffin [6]	97.45	97	97	97
M. Abbasi [13]	93.67	93	93	93
Y. Wang [22]	95.89	95	95	95
Y. Pan [24]	94.12	94	94	94
Proposed	98	98	98	98

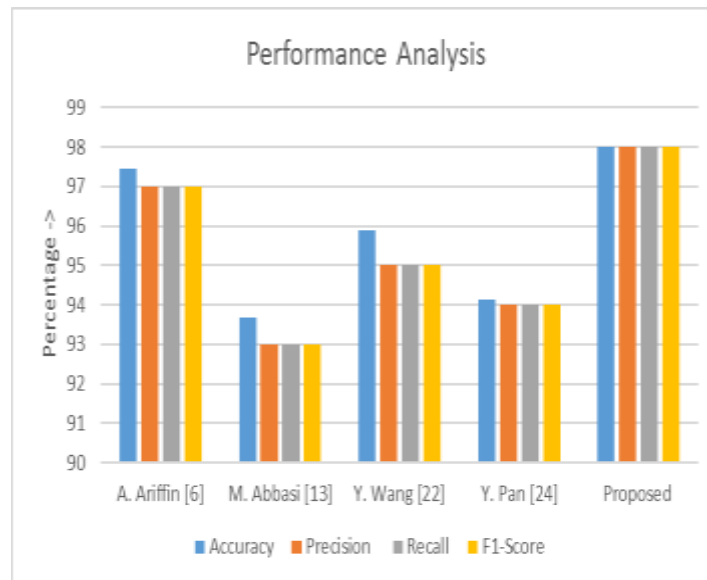


Figure 8. Image List with the Same Label

Performance analysis of proposed model is drawn to the existing models as shown in Figure 8. It is studied that the proposed model comes to as much as 98 percent accuracy, which is relatively 2 percent more than the existing models.

Limitation and Discussion

Although the given model demonstrates a good level of performance, a number of limitations are observed:

- **Dataset Imbalance:** KDD Cup 1999 has very skewed classes (e.g., U2R < 1%), which can introduce a bias to the results. This imbalance can cause biased learning in which the model shows a good performance on the majority classes, but poor performance on minority classes. Consequently, there is a possibility that the model ignores unusual but potentially important attacks, which is unsuitable in the real-world intrusion detection, wherein such attacks can be devastating.
- **Dataset Redundancy:** There are many duplicate records in the dataset which can inflate the performance measures and thereby weaken the generalization capacity. In addition, KDD Cup 1999 is an old data that is gathered more than

twenty years ago. It is not representative of the trends of the modern network traffic, encrypted communication, and modern attack approaches, including botnets, ransomware, and APTs.

- **Computation Cost:** Training is expensive, with demanding GPUs and time-intensive training LSTM units are sequential in nature, and the convolutional multiple layers make the proposed hybrid model computationally inexpensive. The high level of GPUs and memory needed in training, and the long duration of training might not be achievable in organizations with limited resources.
- **Generalizability:** The effectiveness on contemporary network traffic datasets (e.g., UNSW-NB15, CICIDS 2017) should be further validated. Although the model has a high accuracy on the benchmark dataset, the effectiveness of the model in the real-world application setting has not been tested. Network conditions are usually encrypted traffic, adversarial attacks, zero-day attacks, and dynamic protocols that can deteriorate the performance unless managed effectively.

5. Conclusion

With the network traffic growing exponentially, precise predictions of future trends are still significant towards better network management and performance. This study presents a hybrid form of deep learning that combines CNN (Convolutional Neural Networks) and LSTM (Long Short-Term Memory) networks in order to enhance the precision of network traffic prediction. The CNN component is used in extracting features, spatial patterns of the data is identified whereas the LSTM component is good at extracting long-term relationships to identify long-term traffic patterns. The proposed approach allows the use of CNN to extract significant features and LSTM to process sequential data, which is why it surpasses the performance of the traditional deep learning models in the predictive accuracy. The structure is applied to the KDD data and measured with such measures as accuracy, precision and recall. The experimental results show that the model has a significant accuracy of 98 percent, which proves that it is an efficient tool to predict network traffic in real-time and outperforms the current methods. Future research ought to aim at lowering the computational expenses, handling the problem of class imbalance through sophisticated sampling, and testing the model using real-world traffic.

References

- [1] Alam, F., Kashef, R., and Jaseemuddin, M., 2021. Enhancing the performance of network traffic classification methods using efficient feature selection models. In 2021 IEEE International Systems Conference (SysCon), pp. 1-6.
- [2] Poriye, M., and Upadhyaya, S., 2023. Hybrid classification model for network traffic density estimation. In 2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), pp. 1051-1056.
- [3] Adeke, J.M., Chen, J., Zhang, L., Mensah, R.N., and Tong, K., 2020. An efficient approach based on parameter optimization for network traffic classification using machine learning. In 2020 7th International Conference on Dependable Systems and Their Applications (DSA), pp. 99-107.
- [4] Shi, S., Guo, Y., Wang, D., Zhu, Y., and Han, Z., 2023. Distributionally robust federated learning for network traffic classification with noisy labels. *IEEE Transactions on Mobile Computing*, 23(5), pp. 6212-6226.
- [5] Liu, H., and Lang, B., 2021. Network traffic classification method supporting unknown protocol detection. In 2021 IEEE 46th Conference on Local Computer Networks (LCN), pp. 311-314.

- [6] Ariffin, A., Zaki, F., and Anuar, N.B., 2023. Leveraging federated learning and xai for privacy-aware and lightweight edge training in network traffic classification. In 2023 IEEE International Conference on Computing (ICOCO), pp. 47-52.
- [7] Fosić, I., Žagar, D., and Grgić, K., 2022. Network traffic verification based on a public dataset for IDS systems and machine learning classification algorithms. In 2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO), pp. 1037-1041.
- [8] Usama, M., Qayyum, A., Qadir, J., and Al-Fuqaha, A., 2019. Black-box adversarial machine learning attack on network traffic classification. In 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), pp. 84-89.
- [9] Wang, C., Xu, T., and Qin, X., 2015. Network traffic classification with improved random forest. In 2015 11th International Conference on Computational Intelligence and Security (CIS), pp. 78-81.
- [10] Hussain, M.M.K.A., Jaseemuddin, M., and Kashef, R., 2023. Network traffic classification using distributed ML-based data parallelization approach. In 2023 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 539-546.
- [11] Wei, S., Chen, X., and Zhou, B., 2018. Network traffic classification based on AR-ELM algorithm. In 2018 13th World Congress on Intelligent Control and Automation (WCICA), pp. 1500-1505.
- [12] Bader, O., Lichy, A., Dvir, A., Dubin, R., and Hajaj, C., 2024. OSF-EIMTC: an open-source framework for standardized encrypted internet traffic classification. *Computer Communications*, 213, pp. 271-284.
- [13] Geng, Y., Cai, S., Qin, S., Chen, H., and Yin, S., 2021. An efficient network traffic classification method based on combined feature dimensionality reduction. In 2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C), pp. 407-414.
- [14] Wang, X., Chen, S., and Su, J., 2020. App-net: A hybrid neural network for encrypted mobile traffic classification. In IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 424-429.
- [15] Koumar, J., Hynek, K., and Čejka, T., 2023. Network traffic classification based on single flow time series analysis. In 2023 19th International Conference on Network and Service Management (CNSM), pp. 1-7.
- [16] Mishra, S., and Manam, V.C., 2023. A comparative study of unsupervised learning techniques and natural language processing in network traffic classification. In 2023 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 138-143.
- [17] Abbasi, M., Florez, S.L., Shahraki, A., Taherkordi, A., Prieto, J., and Corchado, J.M., 2025. Class imbalance in network traffic classification: an adaptive weight ensemble-of-ensemble learning method. *IEEE Access*.
- [18] Wang, X., Wang, Y., Lai, Y., Hao, Z., and Liu, A.X., 2025. Reliable open-set network traffic classification. *IEEE Transactions on Information Forensics and Security*.
- [19] Lin, W., and Chen, Y., 2024. Robust network traffic classification based on information bottleneck neural network. *IEEE Access*.
- [20] Hu, G., Xiao, X., Shen, M., Zhang, B., Yan, X., and Liu, Y., 2023. TCGNN: packet-grained network traffic classification via graph neural networks. *Engineering Applications of Artificial Intelligence*, 123, 106531.
- [21] Wang, Y., Yun, X., Zhang, Y., Zhao, C., and Liu, X., 2022. A multi-scale feature attention approach to network traffic classification and its model explanation. *IEEE Transactions on Network and Service Management*, 19(2), pp. 875-889.