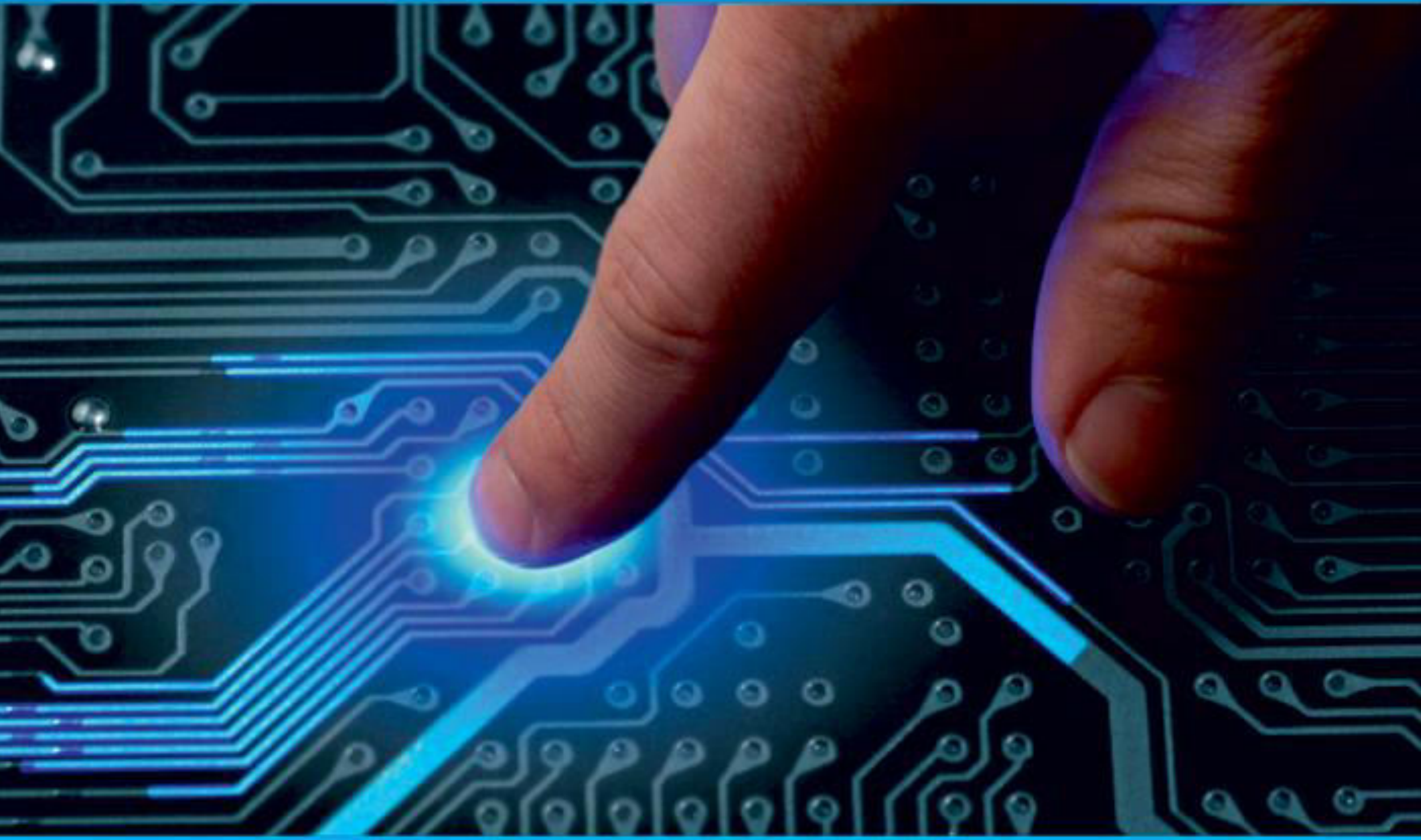
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 10, Issue 6, June 2022

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.165

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006169|

Performance Analysis of Network Anomaly Detection using Bi-LSTM

Prashanthi B, Sonali P, Sowmiya S, Yogalakshmi P, P.B.ARUN PRASAD

UG Students, Department of Computer Science and Engineering, Saranathan College of Engineering, Tiruchirappalli, Tamil Nadu, India

Assistant Professor, Department of Computer Science and Engineering, Saranathan College of Engineering, Tiruchirappalli, Tamil Nadu, India.

ABSTRACT: As use of cloud applications has become more prominent, there is indeed a higher demand for network intrusion detection systems. Network Intrusion Detection (NID) is a vital element of strategic security because of the constantly escalating internet traffic, and a highly effective NIDS is required since novel sorts of threats are constantly emerging. Signature pattern matching system or an AI/ML-based anomaly detection system is the backbone of these intrusion detection systems (IDS). In contrast to AI/ML-based methods, which focus on detecting measures or correlations between collections of features to forecast the occurrence of an attack, pattern matching methods typically have significant False Positive Rates. In order to incorporate learning of the spatial and temporal properties of the data, a deep learning model that combines the distinctive strengths of a convolutional neural network and a bi-directional LSTM is developed in this proposed work, model will be trained and tested using the fully publicly available dataset such as NSL-KDD and UNSW-NB15. The outcome of the model has a low False Positive Rate and a high detection rate.

KEYWORDS: Intrusion Detection System, Machine Learning, Deep learning, Bidirectional LSTM, CNN, SVM

I. INTRODUCTION

Nowadays, in any distributed environment system should be network security into consideration. Intrusion detection systems (IDS) offer an additional layer of security to these networks by identifying illegal infiltration scenarios. IDS fall under the categories of anomaly-based systems and signature-based systems. To improve the security in network Convolution Neural Networks (CNN), Recurrent Neural Network algorithms of deep learning were used ot design the security systems .Due to their high False Positive Rate, these models are currently being researched in real-world applications. The NSL-KDD and UNSW-NB15 datasets will be used to evaluate the proposed model in this paper. In 1999-released NSL-KDD dataset [1] is an improved version of the original predecessor KDD99 dataset. In 2015, the University of New South Wales in Australia released the UNSW-NB15 dataset, which highlighted the inadequacies of the KDD98 and KDD99 data sets, citing the omission of contemporary low footprint attacks from these datasets. This paper suggests a hierarchical model that combines layers of 1D CNN and Bi-LSTM to perform on both of these incredibly distinct datasets.

The Bi-LSTM layers, which are essentially a subcategory of RNNs, those model were trained by different features in dataset to detect type of attacks. CNN is used to learn the spatial/high level features of a dataset. The predictions are made in order to determine whether or not an assault will occur as well as the precise category of the attack.Every day, millions of people and hundreds of thousands of institutions communicate with each other over the Internet. The number of people using the Internet has increased very rapidly over the last two decades, but today it is over four billion and this increase continues rapidly. In parallel with these developments, the number of attacks on the Internet is increasing day by day. For these attacks, there are two basic ways to detect an attack and ensure information security. Signature-based identification and anomaly-based detection. In addition, more than half of today's Internet usage is encrypted using the SSL / TLS (Secure Sockets Layer / Transport Layer Security) protocol, and this percentage is increasing day by day. Signature-based methods do not work effectively with this type of data because you cannot view the contents of the encrypted Internet stream. However, the anomaly-based approach analyzes the data based on common properties such as size, connection time, and number of packets. Therefore, you do not have to look at the content of the message and you can also analyze the encrypted logs. Because of all these benefits, anomaly-based

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006169|

detection techniques are widely used to detect and prevent network attacks. The goal was to contribute to the literature by developing a system that uses deep learning techniques to detect network anomalies quickly and effectively.

II. RELATED WORK

The abnormality could represent a new community risk that has never previously occurred. Network security against illegal access has long been studied, but is sometimes difficult to detect or design model to prevent further security threats. Network attacks are often more versatile as a result of networks' technological innovations and the quick growth of linked devices. The conventional investigative techniques are a novel and flexible way to discover community intrusions and may be applied to any type of community structure. At the ML in community anomaly detection, a detailed assessment is provided. From Supervised Learning (SL) to Reinforcement Learning (RL), every class methods information in a one-of-a-kind style, which results in a massive hole with inside the outcome Supervised, unsupervised or semi-supervised mastering versions are picked primarily based totally on the dataset on hand; the share of labelled information is a key aspect in choosing a version. By contrast, RL is a very one-of-a-kind style, it lets in the version to strive all of the state-motion pairs a good way to pick out the first-rate solution. In addition to the version selection, information fine is the maximum crucial element for anomaly detection, it immediately hyperlinks to the prediction performance. Most of the answers are demonstrated with the aid of using public datasets or with inside the simulation, it will be higher to affirm those fashions with inside the actual community. And the useful resource consumption, including education time and CPU utilization, of the version is hardly ever discussed; this shall additionally be taken into consideration and studied to mirror the efficiency. The future scope is to discover greater for the software of deep mastering strategies with inside the subsequent technology community, including SDN and IoT.

The goal is to build a robust intrusion detection system that helps protect against unauthorized access to network resources. The biggest challenge with previous systems was that these attackers quickly adapted and constantly changed their behaviour to circumvent security mechanisms. Data are supported vector machine (SVM), nearest neighbour method (KNN), logistic regression (LR), Naive Bayes (NB), Multi-Layer perception (MLP), random forest (RF), extra tree classifier (ETC), and it was classified using a Decision Tree (DT)[4]. This classification is done to see if the data is normal or intrusive. Model performance was analysed using a subset extracted from the NSL-KDD dataset. Results of performance of all attack classes were above 99%. Proposed model has an effective prediction rate, reduced computational complexity by removing irrelevant features. The overall overview of the implementation is: Initially the data is pre-processed. Then Identification of the four substantial feature subsets from the model is evaluated. Several machine learning classifiers are employed for testing/training of the data. Pre-processing is where the non-numeric attributes are replaced or removed. The classifiers are used to classify data. Features from the dataset are selected randomly to shrink the dimension of data based on which model was trained. Promising results on DOS attack least promising results on U2L attack. Random Forest, extra tree and decision tree achieved 99% of accuracy. Future work is focused on applying optimization techniques. Ensemble based approaches can be examined. Feature Selection methods can be analysed to lessen computational intricacy. Intrusion detection has to be addressed in the wireless mobile network domain.

As Internet security issues increase, it becomes an integral part of the network intrusion detection system (NIDS) in the IoT environment. In the past, most studies on intrusion detection have been experimented with the KDD CUP99 dataset. However, the KDDCUP99 dataset does not have some typical examples of evaluating NIDS compared to the UNSWNB15 dataset [6]. This article proposes a Support Vector Machine (SVM) [8]. Use new scaling methods for binary and multi-classification experiments [12]. The performance of our method is evaluated based on accuracy, detection rate, and false positive rate. Experimental results show the superiority of the proposed SVM method over other methods. The accuracy of the proposed method reaches 85.99% in the binary classification compared to 78.47% by Expectation-Maximization (EM) clustering. For multiple classifications, the proposed SVM can achieve a test accuracy of 75.77%. This is 6.17% higher than the Naive Bayes (NB) test accuracy[3,5].

III. METHODOLOGY AND DISCUSSION

3.1 *Existing System:*

A brief analysis is done on an existing intrusion detection system developed using Support Vector Machine (SVM). This model is applied on 2 datasets- The KDDCUP99 and the UNSW-NB15. This machine learning algorithm is used for binary classification and the corresponding performance of the machine is evaluated[2,10]. The KDDCUP99 dataset was established almost twenty years ago, hence there is much difference between the normal flow defined by the benchmark datasets and the present normal flow. And also the distribution of the existing benchmark training and testing datasets is different on data types, which will cause the classifier to deviate and lead to decrease in accuracy. This is why the existing system is more focused on the UNSW-NB15 dataset. The UNSW-NB15 dataset contains many new attacks of modern networks. The records in training and testing dataset have no redundancy to ensure the reliability of the evaluations. There are around 44 features in this dataset. Support Vector Machine (SVM) is a supervised learning algorithm which is often used as a classifier in intrusion detection systems[9,13]. This system is developed using a nonlinear log function scaling method instead of the traditional linear normalization in the data pre-processing stage, which is more suitable for the UNSW-NB15 dataset. Considering the data distribution, the log function is selected for pre-processing. And also the kernel functions are used to map low- dimensional space to high dimensional space in the SVM model.

3.2 *Proposed System:*

Keeping in mind the disadvantages of the existing system, the proposed model was established using the deep learning algorithms namely, Convolutional Neural Network (CNN) and Bi-directional LSTM (Long Short Term Memory). The proposed model implements 2 datasets namely, UNSW-NB15 and NSL-KDD. They are implemented separately. Both datasets contain 2 separate CSV files for the training and testing sets. Both subsets are fed as the input to the proposed model. The datasets are initially pre-processed using multiple pre-processing methods namely, One Hot Encoding, Normalization, K-Fold Stratification and Oversampling. This is done to optimize the datasets. Later, these pre-processed datasets are sent to the deep learning model for the detection of network intrusions and for evaluating the performance of the proposed model. After the implementation of the model, the classification reports are generated. This includes the accuracy, the detection rate and the false positive rate of the proposed model. Then the ROC curve and the Confusion matrix are generated for both the training and testing datasets.

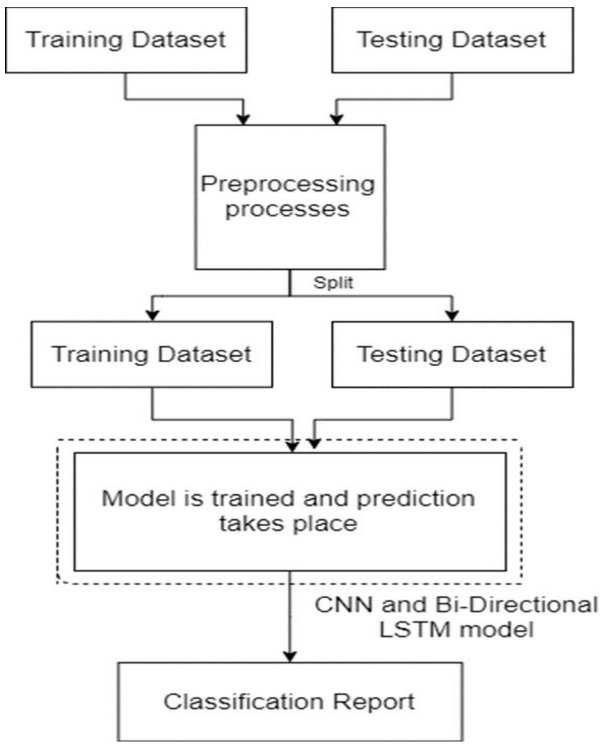


Fig. 1 Proposed Architecture

Fig. 2 represents the proposed model consisting of a 1D CNN layer and several Bi-LSTM layers with a reshaping layer and a batch normalization layer in between. Parameter sharing allows you to reduce the set of parameters and free variables, which reduces the amount of processing resources used for feature extraction. Spatial placement allows previously recognized features to be placed in a sparse matrix so that the correlation between features can be better recognized. Finally, local perception can significantly reduce the training period by reducing the number of parameters. Therefore, 1D-CNN enables rapid spatial learning of specific time series data. The 1D-CNN layer is followed by a maximum pooling layer that allows sample-based discretization of parameters for discovering related functions. This reduces training time and avoids overfitting. After Max Pooling, there is a batch normalization layer. This allows you to normalize the parameters between the middle layers and prevent slow training times. Reshape layer after batch normalization layer reshapes the output of the previous layer for the next pair of Bi-LSTM layers. The Bi-LSTM layer is used to learn from both forward and reverse time series data, and the hidden layer uses two units that have the same input and are connected to the same output. From Fig. 3, it is shown that one of the units processes the forward time series and the other unit processes the reverse time series. This so-called placement aims to increase training time and provide layers with future data that will lead to better functional learning which improves the accuracy of long-span time series data.

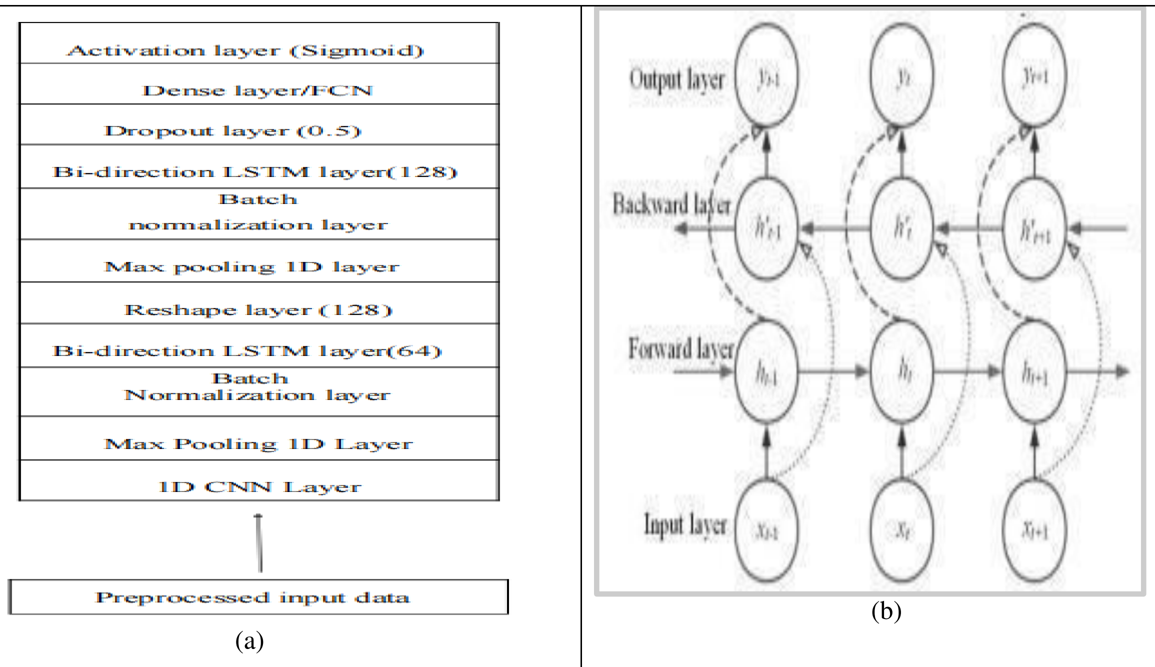


Fig. 2 (a) Layers of the proposed model,(b) Bi-LSTM architecture

The two Bi-LSTM layers in the model are arranged so that the core size doubles with each iteration. According to the block diagram of the model, the first Bi-LSTM layer starts with 64 units and the second last Bi-LSTM layer starts with 128 units. The rationale for this choice is to better understand the correlation of long-range time-varying features learned from the first 1D-CNN layer, mimicking the use of coarse-grained to fine-grained learning and to better feature extraction faster training time. Between each Bi-LSTM layer is a maximum pooling layer that discards the least relevant functions and a batch normalization layer that normalizes the output data of the previous middle layer to improve performance and reduce it training phase. Next, there is a fully connected high density layer. It acts as an exit layer, followed by a dropout layer. The model uses maximum pooling between each layer, while the dropout layer is used to account for over fitting. This is because using a combination of CNNs and RNNs generally increases the likelihood of over fitting and reduces performance on the test set. To check this, the model is evaluated using k-fold cross-validation.

3.3 Implementation

3.3.1 Importing dependencies

The first step before starting the implementation is to import all the required dependencies. Some of the important dependencies are Pandas is required for data manipulation and analysis, Numpy is a library consisting of multidimensional array objects and a collection of routines for performing mathematical and logical operations, Keras acts as an interface for the Tensorflow library and also provides a Python interface for artificial neural networks, LSTM, Simple RNN, GRU, Bidirectional, Batch Normalization, Convolutional1D, MaxPooling1D, Reshape, GlobalAveragePooling1D are the important packages that involve the CNN and Bi-LSTM layers of the model. Sklearn provides a selection of efficient tools for deep learning and statistical modelling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. Tensorflow is a framework created by Google for creating Deep Learning models. Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension Numpy.

3.3.2 Passing datasets as inputs

The datasets namely, UNSW-NB15 and NSL-KDD are fed as inputs to the model. Initially the training and testing set is loaded into the data frame separately. After loading, the columns or features which are not required for the project or the classification are dropped. Now both the sets are checked for any null values which is not recommended.

3.4 Pre-processing

Pre-processing data is an essential step before building a Deep Learning model. When creating a deep learning project, it is not always that we come across clean and well-formatted data. Therefore while doing any operation with the data, it is mandatory to clean it and put it in a formatted way.Data pre-processing is the process of preparing the raw data and making it suitable for a deep learning model and it is also the first and crucial step while creating a model. There are 4 pre-processing methods.

3.4.1 One Hot Encoding

One hot encoding can be defined as the essential process of converting the categorical data variables to be provided to machine and deep learning algorithms which in turn improve predictions as well as classification accuracy of a model. One Hot Encoding is a common way of pre-processing categorical features for deep learning models.

In the datasets, Features with non-numerical instances are initially stored in a separate array. This array is then passed as a parameter to the one hot encoding function. As said, this function helps in converting them into numerical values using the get-dummies function of pandas python library. After one hot encoding both the training and the testing sets are combined as a single dataset.

3.4.2 Normalization

Normalization is the process of re-scaling the data into a particular range to reduce redundancy and improve training time of the model. Min-Max Normalization was discussed in equation (1). For every feature in Min-Max normalization, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

The combined dataset after one hot encoding is passed to the normalization function where the data is normalized. Equation given below represents the formula for Normalization.

$$X[i] = \frac{X[i] - X_{min}}{X_{max} - X_{min}} \qquad (1)$$

3.4.3 K-Fold Stratification

The stratified K fold validation is an extension of the cross-validation technique used for classification problems. It maintains the same class ratio throughout the K folds as the ratio in the original dataset.

The most used validation technique is K-Fold validation which involves splitting the training dataset into k folds. The first k-1 folds are used for training, and the remaining fold is held for testing, which is repeated for K-folds. A total of K folds are fit and evaluated, and the mean accuracy for all these folds is returned. Stratification ensures that each fold is a good representation of the whole dataset, this leads to parameter fine tuning and helps model in classifying the attacks better. This process has shown an optimistic result for balanced classification tasks.

3.4.4 Oversampling

Random Oversampling duplicates data points randomly from the minority classes; this reduces the data imbalance and improves prediction accuracy of minority class. Over sampling python library is used for oversampling with ‘minority’ as parameter.

Over sampling is done only for the UNSW-NB15 dataset. The UNSW-NB15 dataset has an extremely low number of records for categories like Worms, Fuzzers etc. To solve this issue, Oversampling technique has been used in the training set to make sure every attack category has a comparable number of records. NSL-KDD Dataset has a refined number of records with every attack category hence it does not require oversampling.

3.5 Deep Learning Model

After successful pre-processing of the dataset it is then passed to the deep learning model for implementation. This model as said is built with 1D-CNN[11] and Bi-Directional LSTM layers. Initially a sequential model is initialized. The Sequential model is a linear stack of layers. Then as shown in the model architecture block diagram the corresponding layers are added to the model in that exact sequence. After the addition of the layers the model is compiled.

3.5.1 Training phase

The combined dataset is now split according to the number of K-Folds specified. As a result 4 sets are obtained namely, train_x, test_x, train_y and test_y. Using these sets the classification is done. We obtain x_train_1 ,y_train_1 ,x_test_2 and y_test_2. Now the model is trained using the fit function.

The fit method is responsible for numerous operations. Typically, they start by clearing any attributes already stored on the estimator and then perform parameter and data validation. They are also responsible for estimating the attributes out of the input data and store the model attributes and finally return the fitted estimator.

In the fit function, x_train_1 and y_train_1 are passed to train the model x_test_2 and y_test_2 are passed as validation data. Validation data is the data on which to evaluate the validation loss and any model metrics at the end of each epoch. The model will not be trained on this data. And the number of epochs is also passed as a parameter.

In terms of neural networks, an epoch refers to one cycle through the full training dataset. One epoch is when the entiredataset is passed forward and backward through the neural network once. In order to generalize the model, we use more than one epoch in the majority of the deep learning models. The more the number of epochs, the more the parameters is adjusted thus resulting in a better performing model. However, too many epochs might lead to over fitting. If a model is over fitted, it does well in the train data and performs poorly on the test data.

3.5.2 Testing phase

After training the model, predictions are done using the testing dataset. To do so, we need to call the method predict() that will essentially use the learned parameters by fit() in order to perform predictions on new, unseen test data points. Essentially, predict() will perform a prediction for each test instance and it accepts only a single input, x_text_2.

The predicted values of the provided test instances will be returned in the form of an output of an array and it is stored in a variable named pred. Using pred the accuracy values are calculated and stored in a different array called oos_pred after the completion of the epochs.

3.5.3 Classification report

After the predictions, the classification report is generated. This classification report contains the confusion matrix and the ROC curve for both the training and testing datasets separately. A confusion matrix presents a table layout of the different outcomes of the prediction and results of a classification problem and helps visualize its outcomes. A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the deep learning model. This gives us a holistic view of how well the classification model is performing and what kinds of errors it is making. A ROC curve is constructed by plotting the true positive rate (TPR) against the false positive rate (FPR). The true positive rate is the proportion of observations that were correctly predicted to be positive out of all positive observations. Similarly, the false positive rate is the proportion of observations that are incorrectly predicted to be positive out of all negative observations.

IV. RESULTS

A comparative study between the UNSW-NB15 dataset and the NSL-KDD datasets was done and applied on the proposed model.

TABLE 1 COMPARATIVE STUDY ON UNSW-NB15 AND NSL-KDD DATASET

Parameters	**Datasets**	
	UNSW-NB15	**NSL-KDD**
About	UNSW-NB15 is an Intrusion Detection Dataset.	NSL-KDD is a new version of the KDD'99 data set.
Features	It has **45 features.** These features are classified into 6 groups such as 1.Flow 2.Basic 3.Content 4.Time 5.Additional generated 6.Labelled	It has **43 features.**
Instances	Total no. of instances: **2,57,673** Training set: **175,341** Testing set: **82,332**	Total no. of instances: **148,517**

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006169|

Attacks	It contains 1 normal and 9 type of attacks. Namely,
	1.Analysis
	2.Backdoor
	3.DoS
	4.Shellcode
	5.Reconnaissance
	6.Exploits
	7.Fuzzers
	8.Generic
	9.Worms
		It analyses 5 types of attacks. Namely,
		1.Normal
		2.Dos
		3.Probe(Probing attacks)
		4.R2L(Root to Local attacks)
		5.U2R(User to Root attacks)

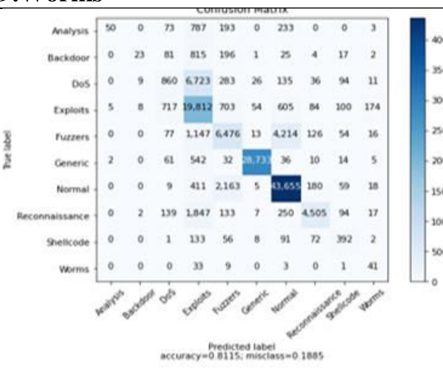
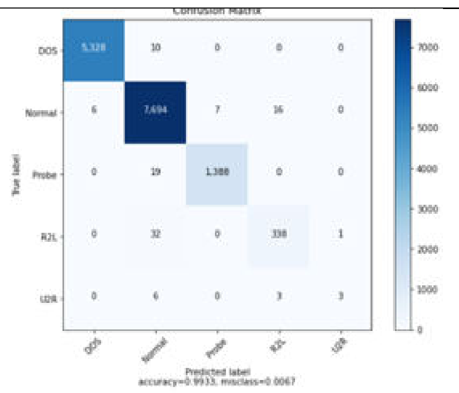
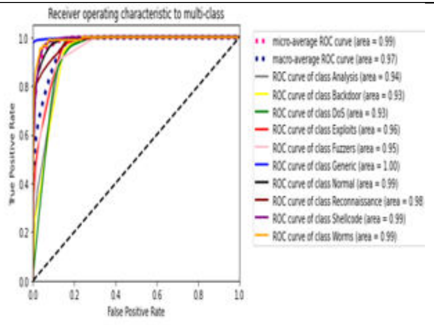
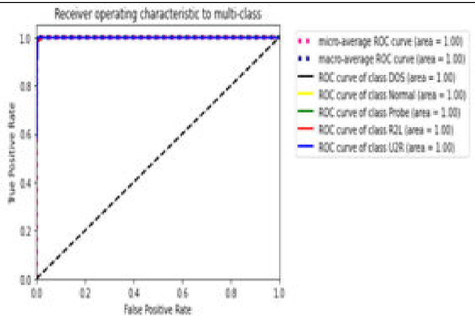
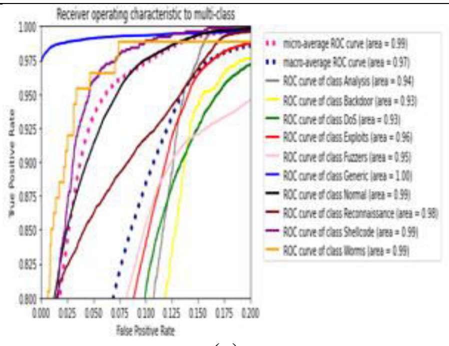
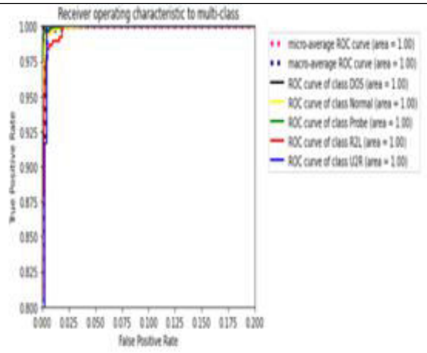
Confusion matrix	 (a)	 (b)
ROC curve	 (c)	 (d)
AUC curve	 (e)	 (f)
Results	**Accuracy: 0.8146** **FPR: 0.040**	**Accuracy: 0.9933** **FPR: 0.003**

Figure 3 (a) Training Confusion matrix ofUNSW-NB15 (b) Training Confusion matrix ofNSL-KDD
(c) ROC of UNSW-NB15 (d) ROC of NSL-KDD (d) AUC of UNSW-NB15 (e) AUC of NSL-KDD

Analysis of the implementation of both the datasets is discussed in TABLE 1. Comparison of UNSW-NB15 and NSL-

KDD is done based on different factors such as no. of features, size of the training and testing datasets, the different attack types detected, Confusion matrix and the ROC-AUC curve obtained for the training and testing sets, accuracy and the false positive rate of the model.

V. CONCLUSION

This paper proposes a model for analysing network traffic containing various variables such as protocol type, service type etc. Oversampling is built in for unbalanced datasets. Combination with CNN and Bidirectional LSTM layer enables learning for spatial and temporal characteristics. The results of the proposed model after training and testing NSL-KDD and UNSW-NB15 records gives promising prospects for real-time use for intrusion detection systems. Since the number of attack classes in the UNSW-NB15 dataset is greater than the attacks in the NSL-KDD dataset, there exists an imbalanced nature and complexity among the attack classes. As a result, it gives an accuracy of 81% for the UNSW-NB15 dataset and 99% for NSL-KDD dataset. The results indicate that there is a need for optimization aimed at the model for U2R and worm category attacks in the future.

REFERENCES

[1] Abrar, I., Ayub, Z., Masoodi, F., & Bamhdi, A. M. (2020, September). "A machine learning approach for intrusion detection system on NSL-KDD dataset." In *2020 International Conference on Smart Electronics and Communication (ICOSEC)* (pp. 919-924). IEEE.

[2] Estévez-Pereira, J. J., Fernández, D., & Novoa, F. J. (2020). "Network Anomaly Detection Using Machine Learning Techniques." In
Multidisciplinary Digital Publishing Institute Proceedings (Vol. 54, No. 1, p.8).

[3] Ingre, B., & Yadav, A. (2015, January). "Performance analysis of NSL-KDD dataset using ANN." In *2015 international conference on* Halimaa, A., & Sundarakantham, K. (2019, April). "Machine learning based intrusion detection system." In *2019 3rd International conference on trends in electronics and informatics (ICOEI)* (pp. 916-920). IEEE.

[4] Jamil, F., & Kim, D. (2021). "An Ensemble of a Prediction and Learning Mechanism for Improving Accuracy of Anomaly Detection in Network Intrusion Environments." Sustainability, 13(18), 10057.

[5] Jing, D., & Chen, H. B. (2019, October). "SVM based network intrusion detection for the UNSW-NB15 dataset." In *2019 IEEE 13th international conference on ASIC (ASICON)* (pp. 1-4). IEEE.

[6] Kocher, G., & Kumar, G. (2020). "PERFORMANCE ANALYSIS OF MACHINE LEARNING CLASSIFIERS FOR INTRUSION DETECTION USING UNSW-NB15 DATASET." *Comput. Sci. Inf. Technol. (CS IT)*, 31-40.

[7] Kim, J., Kim, J., Kim, H., Shim, M., & Choi, E. (2020). "CNN-based network intrusion detection against denial-of-service attacks." *Electronics*, *9*(6), 916.

[8] Mhamdi, L., McLernon, D., El-moussa, F., Zaidi, S. A. R., Ghogho, M., & Tang, T. (2020, October). "A deep learning approach combining autoencoder with one-class SVM for DDoS attack detection in SDNs." In 2020 IEEE Eighth International Conference on Communications and Networking (ComNet) (pp. 1-6). IEEE.

[9] Meftah, S., Rachidi, T., & Assem, N. (2019). "Network based intrusion detection using the UNSW-NB15 dataset." *International Journal of Computing and Digital Systems*, *8*(5), 478-487.

[10] Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2021). "Deep learning for anomaly detection: A review." *ACM Computing Surveys (CSUR)*, *54*(2), 1-38.

[11] Sharma, N., & Yadav, N. S. (2021, September). "Ensemble Learning based Classification of UNSW-NB15 dataset using Exploratory Data Analysis." In 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions (ICRITO) (pp. 1-7). IEEE.

[12] Zavrak, S., &İskefiyeli, M. (2020). "Anomaly-based intrusion detection from network flow features using variational autoencoder." IEEE Access, 8,108346-108358.

[13] Zhang, C., Costa-Pérez, X., & Patras, P. (2022). "Adversarial Attacks Against Deep Learning-Based Network Intrusion Detection Systems and Defense Mechanisms." IEEE/ACM Transactions on Networking.

[14] The UNSW-NB15 dataset can be downloaded from the link:
https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15

[15] The NSL-KDD dataset can be downloaded from the link: https://www.kaggle.com/datasets/hassan06/nslkdd