Efficient Deep CNN-BiLSTM Model for Network Intrusion Detection

Jay Sinha*
jay.x.sinha@gmail.com
Ramaiah Institute of Technology
Bangalore, India

Manollas M
manollasm1997@gmail.com
Ramaiah Institute of Technology
Bangalore, India

Abstract

The need for Network Intrusion Detection systems has risen since usage of cloud technologies has become mainstream.With the ever growing network traffic, Network Intrusion Detection is a critical part of network security and a very efficient NIDS is a must, given new variety of attack arises frequently. These Intrusion Detection systems are built on either a pattern matching system or AI/ML based anomaly detection system. Pattern matching methods usually have a high False Positive Rates whereas the AI/ML based method, relies on finding metric/feature or correlation between set of metrics/features to predict the possibility of an attack. The most common of these is KNN, SVM etc., operate on a limited set of features and have less accuracy and still suffer from higher False Positive Rates. In this paper, we propose a deep learning model combining the distinct strengths of a Convolutional Neural Network and a Bi-directional LSTM to incorporate learning of spatial and temporal features of the data. For this paper, we use publicly available datasets NSL-KDD and UNSW-NB15 to train and test the model. The proposed model offers a high detection rate and comparatively lower False Positive Rate. The proposed model performs better than many state-of- the-art Network Intrusion Detection systems leveraging Machine Learning/Deep Learning models.

CCS Concepts: • **Computing methodologies** → **Neural networks**; **Supervised learning by classification**; *Cross-validation*; • **Security and privacy** → **Intrusion detection systems**; Denial-of-service attacks.

Keywords: Intrusion Detection, Deep Learning, Network Traffic, CNN, RNN, Bi-LSTM, LSTM

*Both authors contributed equally to this research.

AIPR 2020, June 26–28, 2020, Xiamen, China

ACM ISBN 978-1-4503-7551-1/20/06...$15.00
https://doi.org/10.1145/3430199.3430224

ACM Reference Format:
Jay Sinha and Manollas M. 2020. Efficient Deep CNN-BiLSTM Model for Network Intrusion Detection. AIPR'20, September 25-27, 2020, Xiamen, China,9 pages

1 Introduction

With the disruptive adoption of cloud technologies since the start of the mainstream Internet usage, the number of Intrusion incidents has also risen exponentially. Since, one data center maintained by any company like Microsoft, Amazon, Google etc hosts a multitude of on-demand servers, platforms etc to provide services to a vast range of small, medium or large enterprises, the cost associated with network security, firewalls has also seen growth incorporating a wide range of techniques for Prevention and Incident Handling to secure data and prevent disruption of services. These intrusions include Eavesdropping, network viruses, probing attacks etc.

Prediction Models based on network time series data, is one of the techniques being used for Network Intrusion Detection Systems[17]. Majority of time-series data has non-linear characteristics due to various data points that change throughout the time because of irregular fluctuations. Several statistical Machine Learning techniques like k-Nearest Neighbours, Support Vector Machine, Naive Bayes etc [7, 12, 14, 16, 21, 24] have been used for NIDS as well. These statistical techniques do not include mutual relations between data, and mostly rely on feature engineering or feature selection, which makes them ineffective for real-time usage with even lower Detection Rates.These models are still under research for practical usage due to their high False Positive Rate.[10] Currently, there has been a widespread usage of deep learning techniques like Convolution Neural Networks (CNN), Recurrent Neural Network etc.

Two datasets will be used for Evaluation of the proposed model in this paper: NSL-KDD [1] and UNSW-NB15 [2] dataset. NSL-KDD is a refined version of the original predecessor KDD99 dataset which was released in 1999 [6]. UNSW-NB15 dataset was published by University of New South Wales, Australia in 2015 which marked the limitations of KDD98 and KDD99 data sets including the fact that these datasets do not include modern low footprint attacks.[19]

To perform on both of these very unique datasets, this paper proposes a hierarchical model by combining layers of

1D-CNN and Bi-LSTM. CNN is used to learn the spatial/hgh-level features of a dataset and the Bi-LSTM layers (essentially a sub-category of RNN) to learn the long time-range temporal features of the data and combine these to predict attacks. The predictions are done for Binary Classification of predicting whether or not an attack is happening and Predicting the exact category of the attack. For multi-category attack prediction, in NSL-KDD, the analysis is done on 5 classes - Normal, Denial of Service (DoS), Probe (Probing Attacks), R2L (Root to Local Attacks) and U2R (User to Root Attack). For multi-category attack prediction in UNSW-NB15, 10 classes have been used: Normal, DoS, Exploits, Generic, Reconnaissance, Worms, Shellcode, Analysis, Backdoor and Fuzzers.

2 Related Work

Deep Learning approaches have always been popular with Network Intrusion Detection problems. With the KDD-99 cup data in circulation, the issue has met with proposed deep learning models and the solutions they provide in an incremental fashion. At first, the approaches solving the same, were focused towards pattern recognition, with [29], discussing the approach with a BM pattern matching algorithm which proves to be accelerated in terms of time performance and recognition speed. After pattern recognition algorithms, researchers have used feature selection by leveraging machine learning and deep learning techniques.

2.1 Machine Learning Techniques:

Traditional machine learning techniques such as Support Vector Machine (SVM) [3], Random Forest [30] and Adaptive Boosting [9] have been often used by researchers for constructing Network Intrusion Detection classifiers. Researchers have also used k-Mean Clustering [20] for efficient classification but they have always turned out to be weak because of high False Positive Rate (FPR), overfitting and with lower accuracy on classes having less percentage of data available as compared to the classes in sufficient numbers. The reason being, the traditional machine learning approaches concentrate upon learning feature importance, feature availability and dimensionality reduction techniques to find the most optimum correlation between data points that seem to have the most amount of influence upon the end-result while completely overlooking the importance of correlation between the features and take into account the time-steps in order to predict the best possible result. This led to the adoption of deep learning approaches in order to resolve the lacking points of the above.

2.2 Deep Learning Techniques:

There is a large number of research made on Network Intrusion detection leveraging Deep Learning techniques.

2.2.1 [26] Using RNN .

NSL-KDD and UNSW-NB15 datasets have time step column which makes the use of RNN a default choice. Recurrent neural networks are an extension to Artificial Neural Networks and are mainly used for analysing time series data to learn long range temporal features. RNNs contain an internal feedback loop in order to store time associations and end up forming an acyclic graph as a result. Backpropagation results in a vanishing gradient problem [5](the features learned at the very start of the network start to have the very least amount of effect on the end-result of the model). To solve this problem LSTM (Long Short Term Memory)[8] and GRU (Gated Recurrent Units) are used in order to solve the vanishing gradient problem. Fig.1 shows the structure of a LSTM cell.

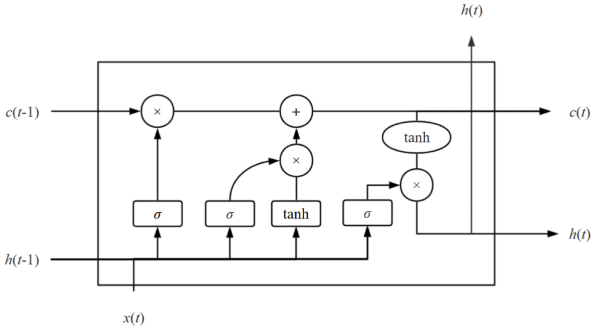


Figure 1. Structure of a LSTM Cell

$$f_t = (W_f.[h_{t-1}, x_t] + b_f), \tag{1}$$

$$i_t = tanh(W_i[h_{t-1}, x_t] + b_i), \tag{2}$$

$$C_t = tanh(W_C.[h_{t-1}, x_t] + b_C), \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * \ C_t, o_t = (W_o[h_{t-1}, x_t] + b_o, \tag{4}$$

$$h_t = o_t * tanh(C_t), \tag{5}$$

where f, i, t, o, h, C, W, b denote forget, input, time-step, output layer, hidden layer, cell state, Weight matrix, bias respectively.

In [26], the authors have ran a simulation on a simple RNN, LSTM and GRU and benchmarked them on the KDD'99 Dataset for multi-category predictions and end up achieving accuracy ranging from 94.2% to 96.98% 94.3% to 95.37% on different combination of LSTM layers and GRU layers respectively. On multicategory UNSW-NB15, authors of [26] have reported 64.8% accuracy from a GRU network and 67.5% from a LSTM Network.

2.2.2 [4, 11] Using CNN .

Convolutional Neural Networks or CNN have been the top performer in Image Recognition for learning boundary regions of different objects in an image which are called spatial features.[15]

In [4], the authors propose a CNN with three hidden layers consisting of a Convolutional Layer along with pooling layer and leverages coarse grained to fine grained learning for deepening the network architecture with more post-convolution kernels resulting in mapping of the features in a high-dimensional space to leverage improved learning. The model mentioned is applied to KDD'99 dataset and gives an accuracy of 99.23%.

In [11], the authors have proposed a 1D-CNN citing that 2-D CNNs are mainly efficient with 2-D Images and 1-D CNN can be used better for learning features on a time-series dataset by by serializing TCP/IP packets in a predetermined time range for effective classification. The dataset used in the papers is UNSW-NB15 and the authors have reported an accuracy of 91.2% with 3 layers of 1-D CNN for binary classification on UNSW dataset.

2.2.3 [27] Using CNN and RNN. CNN and RNN are two very different types of neural networks that are being used recently upon time-series data itself. The idea behind combining them is that CNN are used for learning spatial features by increasing the number of kernels which makes learning coarse grained (at the start of the network) and fine-grained (at the end of the network). The RNN in the model learns the temporal features from long range time-series data.

In [27], the authors have proposed such a hierarchical model to leverage spatial and temporal features learning of CNN and RNN respectively. The model consists of multiple CNN layers followed by 2 LSTM layers after preprocessing the dataset. The DARPA and ISCX2012 dataset are very similar to KDD99 Dataset which have attacks combined into 5 upper-level categories and give accuracy of around 50% for DARPA and around 97% for ISCX2012 dataset.

2.2.4 [23, 28] Using BiLSTM . A BiLSTM or Bidirectional LSTM is a type of LSTM network in which the learning data is fed from start to end of the model while feeding the information from end to start as well which allows for better learning at every timestep of data. This results in better learning of features per time step. In [23], the authors have proposed a model called BAT which combines Bi-LSTMs with an Attention layer to filter out features that have the least/minimal impact on the end result of the model. The model consists of multiple convolutional layers with a BiLSTM, Attention, Fully Connected Dense Layer. The model is run on NSL-KDD dataset with its 5 five classes resulting in accuracy ranging from 82.97% to 84.24% on multicategory analysis.

In [28], the author has proposed a two Bi-LSTM layered model with sigmoid activation and has performed analysis on UNSW-NB15 dataset with an average F1-Score of 0.86.

3 EXPERIMENT

In this paper, we propose a model combining 1-Dimensional Convolutional Neural Network (1-D CNN) [25] and multiple layers of Bi-directional LSTM (Bi-LSTM) [22]. In this section, the layers of the proposed model's neural network will be discussed along with the Datasets and the preprocessing techniques used on these datasets.

3.1 Model Architecture

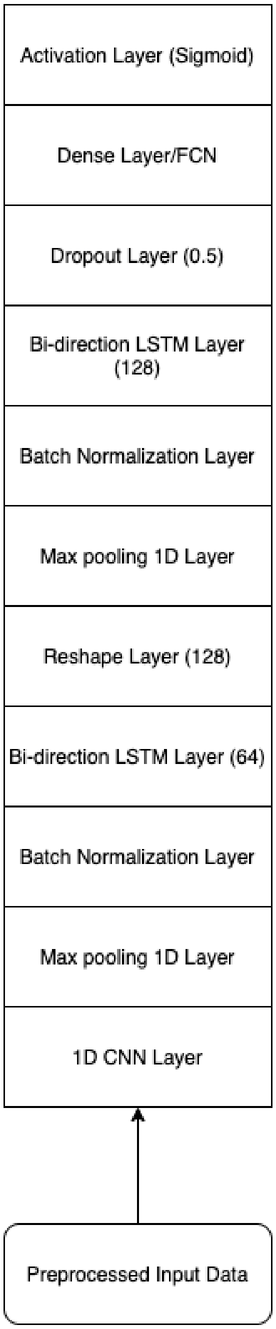

Activation Layer (Sigmoid)
Dense Layer/FCN
Dropout Layer (0.5)
Bi-direction LSTM Layer (128)
Batch Normalization Layer
Max pooling 1D Layer
Reshape Layer (128)
Bi-direction LSTM Layer (64)
Batch Normalization Layer
Max pooling 1D Layer
1D CNN Layer
Preprocessed Input Data

Figure 2. Block Diagram of the model used

As evident from the block diagram of Fig.2, the proposed model consists of a 1-D CNN Layer along with multiple layers of Bi-LSTM with Reshape and Batch Normalization layers in between. The idea is to leverage the 1-D CNN layer and max pooling layer for it's parameter sharing, spatial arrangement and local perception characteristics. Parameter Sharing allows for a reduced set of parameters and free variables that results in feature extraction with fewer use of processing resources. Spatial arrangement allows for the arrangement in a sparse matrix of features recognised so far to enable better recognition of correlation between features. Lastly, local perception allows for reduced number of parameters and hence, decreases the training duration by a huge amount. Therefore, 1-D CNN allows for fast paced spatial learning for the given time-series data. The 1-D CNN layer is followed by a Max Pooling layer which allows for sample-based discretization of parameters in order to recognise the relevant features resulting reduced training time and preventtion from overfitting. After Max Pooling, comes Batch Normalization layer which enables normalization of parameters between intermediate layers to prevent slower training times. Reshape Layers help reshape the output of the previous layer in between the two Bi-LSTM layers.

Bi-LSTM layers are used to learn from both forward and backward time series data with the hidden layers making use of two units having the same input and connected to the same output. The Structure of a Bidirectional LSTM Cell can be seen in Fig.3. One of the unit processes forward time series and the other processes backward time series. This so called arrangement is said to provide the layers with future data for boosting training time with better learning of features resulting in greater precision for a long spanning time-series data.

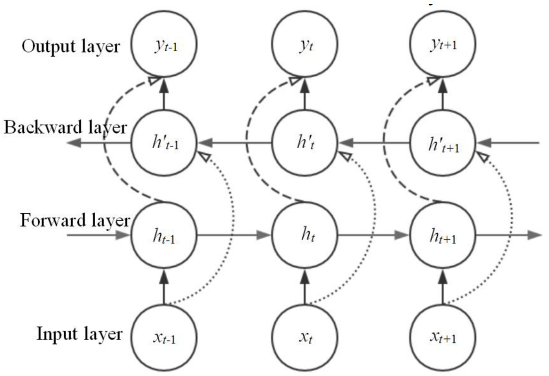


Figure 3. Structure of a Bidirectional LSTM Cell

The two Bi-LSTM layers in the model are arranged in a manner which doubles it's kernel size in every iteration. As per the block diagram of the model, the first Bi-LSTM layer starts with 64 units and the next and last Bi-LSTM layer 128 number of units. The reason for this choice is to mimic the use of coarse grain to fine grained learning to better understand the correlation of long range time dependent features which provides for better extraction of features and faster training times. Between each Bi-LSTM layer, there is a Max Pooling layer to dismiss the least relevant features and the Batch Normalization layers to normalize the output data of the previous intermediate layer in order to boost performance and decrease training times.

The Dropout Layer is put in place to account for Over Fitting even though the model uses Max Pooling in between every layer. The Fully Connected Dense layer comes next which serves as an Output Layer. The reason behind this is that, generally, CNN and RNN used in combination have a higher probability of over fitting and perform poorly on the testing set. To keep this in check, the model is evaluated with k-fold cross validation.

3.2 Dataset

The proposed model in this paper, is evaluated on two datasets: NSL-KDD and UNSW-NB15.

3.2.1 NSL-KDD Dataset.

The NSL-KDD dataset was made available by University of New Brunswick. The NSL-KDD Dataset is an improvement on KDDCup'99 Dataset as the latter one has inherent drawbacks revealed by various analyses. NSL-KDD contains the essential records of the complete KDD Dataset and has been one of the most popular dataset for Network Intrusion Detection Systems analysis. NSL-KDD has a lot distinctions from its predecessor including: Removal of redundant records [6], sufficient availability of records in training, testing dataset and number of selected records from each difficulty group inversely proportional to percentage of records in the original KDD Dataset.

Table 1. NSL-KDD Dataset Attack Categories

Category	Count
Normal	77054
DoS	53385
Probe	14077
R2L	3749
U2R	252
Total	148517

3.2.2 UNSW-NB15 Dataset.

This dataset was published by University of New South Wales in 2015. Since its inception, UNSW dataset has been widely used, it includes a wider variety of attacks families, number of features extracted, number of distinct IP addresses used for simulation and collection of data [19]. This data set consists of a hybrid of the real modern normal and the contemporary synthesized attack activities of the network traffic. Table 1 and 2 shows a list of features available in NSL-KDD and UNSW-NB15 Datasets.

Table 2. UNSW-NB15 Dataset Attack Categories

Category	Count
Normal	93000
Analysis	2677
Backdoor	2329
DoS	16353
Exploit	44525
Fuzzers	24246
Generic	58871
Reconaissance	13987
Shellcode	1511
Worms	174
Total	257673

3.3 Pre-Processing

Preprocessing of the datasets is generally handled by Normalization of numeric features and One Hot Encoding of Categorical features for both NSL-KDD and UNSW-NB15 dataset. But as discussed earlier, NSL-KDD Dataset has a refined number of records with every attack category. On the other hand, 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.

3.3.1 One Hot Encoding. There are categorical features in both NSL-KDD and UNSW-NB15 datasets and these should be converted to numerical values for our deep learning model to give out good prediction results.Hence these columns have been converted into numerical values in the pre-processing part using get-dummies function of pandas python library. One-hot encoding is chosen over label encoder since label encoder will produce multiple numbers in the same column, the model might misunderstand these values to be in a particular order and this will impact the classification.

3.3.2 Normalization. Normalization is rescaling the data into a particular range to reduce redundancy and improve training time of the model. Min-Max Normalization is used in the paper and rescales the range of the data to [0,1].

$$X[i] = \frac{X[i] - X_{min}}{X_{max} - X_{min}} \tag{6}$$

3.3.3 Stratified K-cross fold validation. Stratification is the process of rearranging the data to ensure each fold is a good representative of the whole. Stratified K-cross fold validation technique splits the dataset into K sets and the model uses K-1 folds for training and is validated on the Kth fold. This is continued until all the folds are used to validate the model once. 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.K-cross fold method is chosen over other validation methods since it performs better than other methods and requires less computation power[13].

3.3.4 Oversampling. Random Oversampling duplicates data points randomly from the minority class, this reduces the data imbalance and improves prediction accuracy of minority class. RandomOverSampler class of imblearn. oversampling python library is used for oversampling with 'minority' as parameter. The number of samples of minority class Worms in UNSW-NB15 dataset is 173 this is very little compared to the total number of samples 257,673 and this imbalance decreases the prediction accuracy of the minority class[18]. Hence random oversampling technique is applied only on the training set of UNSW-NB15 and there is a significant increase in accuracy, detection rate of the class Worms as you can see in the Fig.11.

4 Evaluations and Discussion

4.1 Evaluation Metrics

Some of the metrics that are used to evaluate the performance of the proposed model are Accuracy(ACC),Detection Rate(DR), False Positive Rate(FPR), F1-Score and ROC-AUC curve.

Accuracy and DR measures the model's ability to predict all classes and attacks respectively. FPR is the percentage of normal records classified as attacks and this is a very important metric along with DR and ACC. If the FPR is high then the model may not be effective although it has good DR and ACC. F1-score gives more realistic measure of the performance as precision and recall may not give a clear picture of the performance alone. The definition of the above mentioned metrics are given in Equations (7),(8),(9) and (10).

$$Accuracy(ACC) = \frac{TP + TN}{TP + TN + FP + FN}, \tag{7}$$

$$DetectionRate(DR) = \frac{TP}{TP + FN}, \tag{8}$$

Where TP is the number of attacks correctly classified, TN is the number of normal traffic correctly classified,FN is the number of attacks misclassified as normal traffic and FP is the number of normal traffic misclassified as attack.

$$FalsePositiveRate(FPR) = \frac{FP}{FP + TN}, \tag{9}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall}, \tag{10}$$

Finally the ROC-AUC curve measures the model's capability of distinguishing between classes of the dataset when the threshold is varied. AUC (Area Under Curve) is the entire area underneath the ROC curve and it is a value that varies between 0 to 1. Higher the AUC, better the model is at classifying different classes correctly.

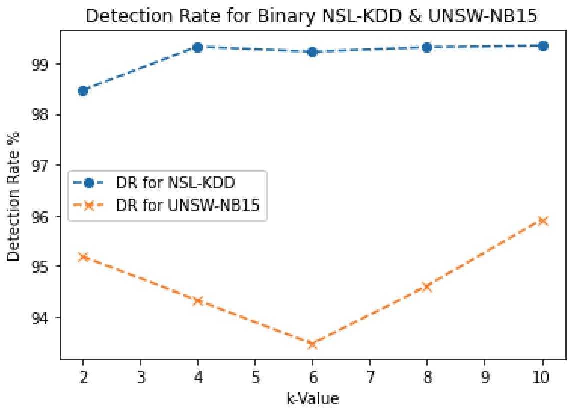


Figure 4. Detection Rate Plot of Binary Classification for UNSW-NB15 and NSL-KDD dataset

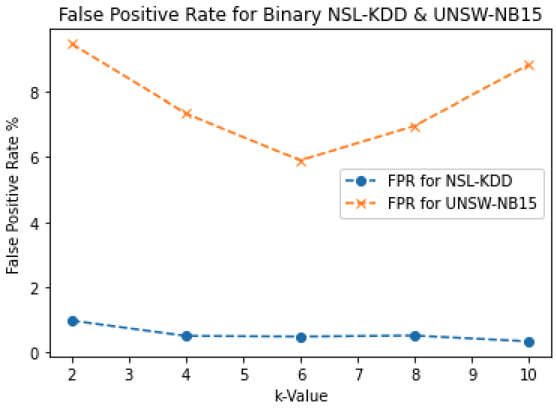


Figure 5. False Positive Rate Plot of Binary Classification for UNSW-NB15 and NSL -KDD dataset

4.2 Model Results

4.2.1 Binary Classification. In Binary Classification the model predicts whether the sample is an attack or belongs to normal class. Table 3 has the results of binary classification by the proposed model under different Stratified K-Fold Cross Validation for k=2 to 10, the average detection rate (DR%) for UNSW-NB15 dataset is 94.70%, accuracy (ACC) is 93.84% and false positive rate (FPR%) is 7.70%. The model shows high detection rate (DR%) and comparatively low false positive rate (FPR%) and these results for various k-values are plotted in Fig.4, Fig.5 respectively . Fig.8 shows different F1-Scores for k ranging from 2 to 10, the best F1-Score for binary classification is 0.9548 for k=10. The maximum accuracy is 94.21% and detection rate is 95.92% which is obtained when k is 10 and correctly so because as the number of folds increase there will be more sample of each attack/normal class available for the model to train and hence the model will be able to classify them better.

Table 3. Result for Binary Classification

K	NSL-KDD			UNSW-NB15		
	ACC%	DR%	FPR%	ACC%	DR%	FPR%
2	99.00	98.47	0.98	93.51	95.19	9.47
4	99.27	99.33	0.51	93.73	94.33	7.34
6	99.37	99.23	0.49	93.70	93.47	5.90
8	99.40	99.32	0.52	94.07	94.61	6.96
10	99.50	99.35	0.34	94.21	95.92	8.83
Average	99.30	99.14	0.56	93.84	94.70	7.70

Table 4. Result for Multi-Class Classification

K	NSL-KDD			UNSW-NB15		
	ACC%	DR%	FPR%	ACC%	DR%	FPR%
2	98.90	98.35	0.54	81.30	92.40	6.67
4	99.13	98.83	0.50	82.03	92.34	6.08
6	99.33	99.06	0.33	82.32	92.39	5.50
8	99.36	99.13	0.37	82.32	93.37	7.00
10	99.40	99.04	0.42	82.45	92.09	5.21
Average	99.22	98.88	0.43	82.08	92.51	6.09

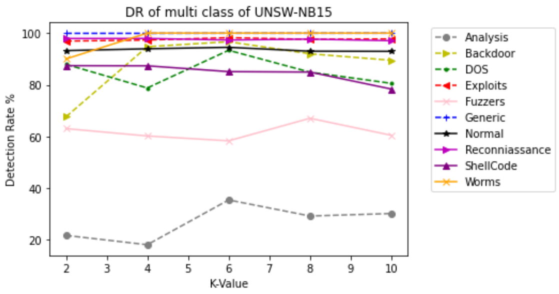


Figure 6. Detection Rate Plot of all classes for UNSW-NB15

In binary NSL-KDD, the model gives an average accuracy of 99.30% for k-value ranging from 2 to 10 with the best accuracy of 99.50% for. The average Detection rate given by model is 99.14%. While the model increases DR from k-value 2 to 4, a decrease in the DR can be seen for k-value = 6. The

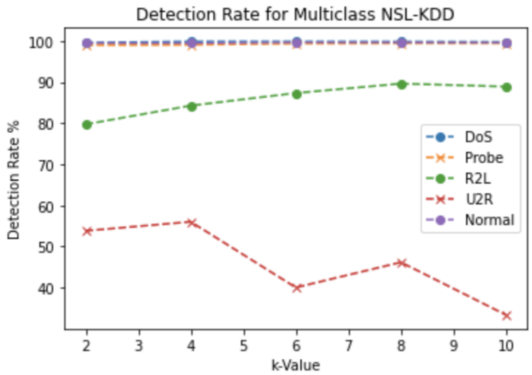


Figure 7. Detection Rate Plot of all classes for NSL-KDD

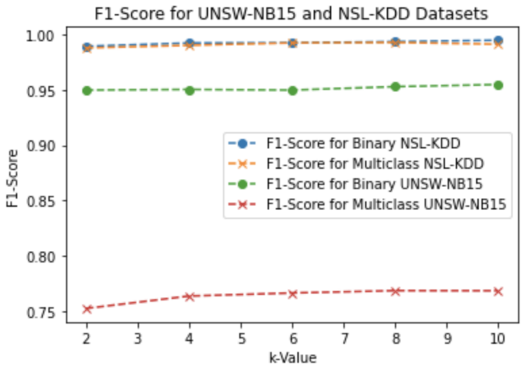


Figure 8. F1-Score Plot for NSL-KDD and UNSW-NB15 Datasets

.

average FPR% given by the model is 0.56 with the least FPR received for k-value = 10. The individual and plots metrics as per k-value can be seen in Fig.4, Fig.5

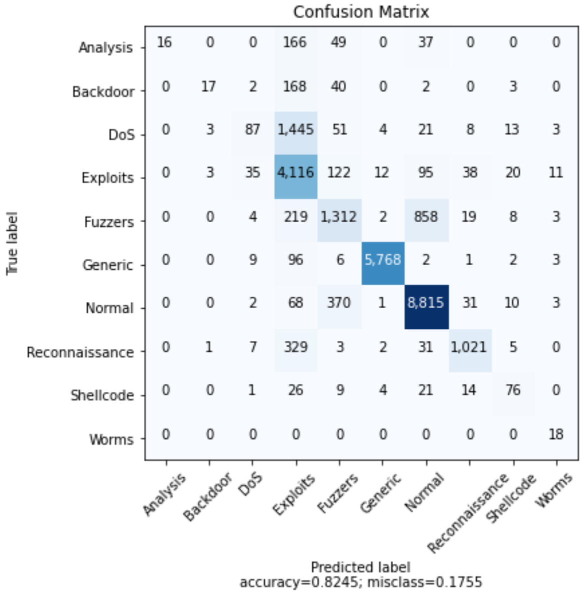
F1-Score from Fig.8 for binary NSL-KDD analysis, shows a rise in values from k-value 2 to 10 with the best value of 0.9949.

4.2.2 Multi-Class Classification.

The results of multi class classification on UNSW-NB15 Dataset for k ranging from 2 to 10 can be seen in Table 4. The average accuracy is 82.08% (ACC), detection rate (DR%) is 92.51% and false positive rate is 6.09%.The best F1-Score of multi-class classification is 0.7684 for k=8 and F1-Score for different k values is given in Fig.8.

Detection rate of each of the 10 classes is plotted in Fig.6 ,the model is able to classify Normal,Backdoor,DOS,Exploits, Shellcode, Generic, Reconnaissance and worms very accurately. The model has an average capability to detect the class Fuzzers. The FPR% is comparitavely low and better than any state-of-the-art models and it is seen in Table 4.

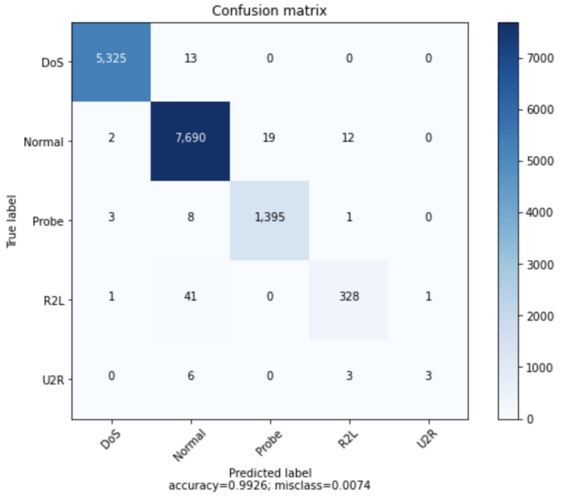
The class Analysis has very low DR% because this class has 2677 samples in the dataset, this is just 1.03% of the total samples in the dataset and the model does not have enough data to perform better on this class. The class Worms has a lower percentage of records(0.067%) than the class Analysis but the detection rate of the class Worms is reaching 100% as plotted in the Fig.11, this is the result of oversampling. The model was able to train better since more samples were available for training as oversampling was applied on the class Worms. In the confusion matrix of Fig.9 all 18 records in the test set were classified as Worms and detection rate is 100% and FPR% is 0.


Confusion Matrix

True label	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Normal	Reconnaissance	Shellcode	Worms
Analysis	16	0	0	166	49	0	37	0	0	0
Backdoor	0	17	2	168	40	0	2	0	3	0
DoS	0	3	87	1,445	51	4	21	8	13	3
Exploits	0	3	35	4,116	122	12	95	38	20	11
Fuzzers	0	0	4	219	1,312	2	858	19	8	3
Generic	0	0	9	96	6	5,768	2	1	2	3
Normal	0	0	2	68	370	1	8,815	31	10	3
Reconnaissance	0	1	7	329	3	2	31	1,021	5	0
Shellcode	0	0	1	26	9	4	21	14	76	0
Worms	0	0	0	0	0	0	0	0	0	18

Predicted label
accuracy=0.8245; misclass=0.1755

Figure 9. Confusion Matrix for UNSW-NB15


Confusion matrix

True label	DoS	Normal	Probe	R2L	U2R
DoS	5,325	13	0	0	0
Normal	2	7,690	19	12	0
Probe	3	8	1,395	1	0
R2L	1	41	0	328	1
U2R	0	6	0	3	3

Predicted label
accuracy=0.9926; misclass=0.0074

Figure 10. Confusion Matrix for NSL-KDD

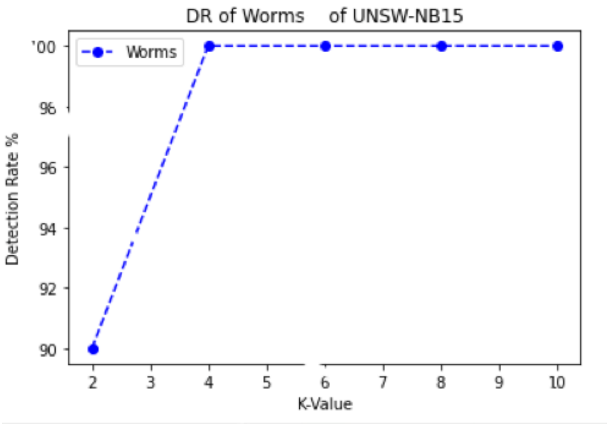


Figure 11. Detection Rate Plot of Worms attack class for UNSW-NB15

Fig.12 is ROC-AUC plots on UNSW-NB15, the Area Under Curve (AUC) of all classes are between 0.94-1.0 and the average AUC is 0.971. This is an indicator that the model is very efficient and accurate in distinguishing among various classes of the dataset.

For NSL-KDD dataset in multiclass, the model gives an average Accuracy of 99.22% with the best accuracy of 99.4%. The average Detection Rate is 98.88% with the best result of 99.13%. The average FPR% is 0.43 with the best value of 0.33. Examining the plot of individual class DR in Fig.7 shows decreasing value of DR for U2R category for k-values greater than 4 which is due to the fact that the training set has least number of U2R samples. Three classes including Normal, DoS, Probe have a high DR. The individual values for Accuracy, DR and FPR along with plots can be checked in Table 4. The confusion matrix can be seen in Fig10.

F1-Score for multiclass NSL-KDD analysis, shows a rise in values from k-value 2 to 8 with a slight dip at k = 6. As discussed, the F1-Score is a very reliable metric for testing a model, which is 0.9929. The plot can be seen in Fig.8.

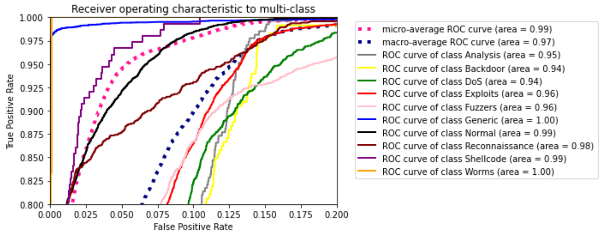


Figure 12. ROC-AUC Curve for UNSW-NB15

Fig.13 illustrates the AUC for multiclass NSL-KDD which is 1.00 for all classes. As mentioned before, AUC is the measure of model's capability to distinguish between different classes/categories inside a dataset for which the model performs well on NSL-KDD dataset.

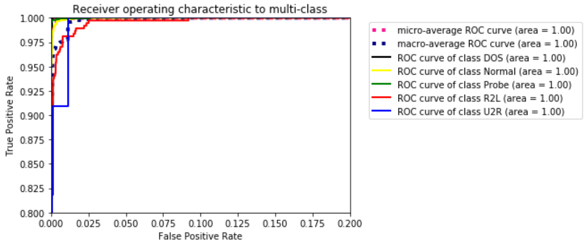


Figure 13. ROC-AUC Curve for NSL-KDD

4.3 Comparison with other models

4.3.1 NSL-KDD.

Comparing the proposed model with other models like: Few-Shot Learning, Bi-LSTM Attention (BAT) model, HAST-IDS and SVM. In comparison, it is clear that the proposed model offers improved performance across metrics specially in Detection Rate and can be seen in Table 5. In comparison, the FPR is lower in BAT but still when taking into account all factors including DR and Accuracy, the proposed model becomes the favourable choice. The closest model in overall performance is HAST-IDS model as it provides little lower results.

Table 5. NSL-KDD Multiclass Comparison

Model	DR%	FPR%	Accuracy%
Proposed Model	98.882	0.43	99.22
Few-Shot Learning	92.06	4.22	92.33
BAT-MC Model	83.6	0.34	84.15
HAST-IDS	95.85	-	93.27
SVM	-	-	69.52
1-D CNN	-	-	78.97

4.3.2 UNSW-NB15.

Comparing the results of proposed model for UNSW-NB15 dataset in multiclass with other models like Adaboost, LSTM, SVM. As evident from Table 6, the proposed model gives better performance across all metrics including Detection Rate, False Positive Rate and Accuracy. The closest model in this comparison is HAST-IDS as it has a 1.2% higher Detection rate but larger False Positive Ratio along with lower accuracy which makes the proposed model more preferable for use in Intrusion Detection Systems.

Table 6. UNSW-NB15 Multiclass Comparison

Model	DR%	FPR%	Accuracy%
Proposed Model	92.506	6.092	82.084
Adaboost	91.13	22.11	73.19
LSTM	92.06	4.22	92.33
SVM	83.71	7.73	74.8
HAST-IDS	93.65	9.6	80.03

5 Conclusion

This paper proposes a model for analysing network traffic including a multitude of variables like protocol type, service type etc. Oversampling was incorporated in order to account for imbalanced datasets. The combination of using CNN and Bi-directional LSTM layers enabled learning of spatial and temporal features. Result of the proposed model after training and testing on NSL-KDD and UNSW-NB15 datasets gives promising prospective real-time usage for Intrusion Detection systems. As evident from the result, the need to optimize the model for U2R category of attacks is to be investigated in the future to allow for testing in a honeypot system.

References

[1] [n.d.]. http://nsl.cs.unb.ca/NSL-KDD/

[2] [n.d.]. https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/

[3] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim. 2018. Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection. *IEEE Access* 6 (2018), 33789–33795.

[4] M. Azizjon, A. Jumabek, and W. Kim. 2020. 1D CNN based network intrusion detection with normalization on imbalanced data. In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. 218–224.

[5] Y. Bengio, P. Simard, and P. Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks* 5, 2 (1994), 157–166.

[6] L. Dhanabal and Dr. S.P. Shantharajah. 2015. A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms. *International Journal of Advanced Research in Computer and Communication Engineering* 4. Issue 6.

[7] S. Garg and S. Batra. July 2017. A novel ensembled technique for anomaly detection. *Int. J. Commun. Syst.* 30, 11 (July 2017), 32–48. https://doi.org/10.1002/dac.3248

[8] S. Hochreiter and J. Schmidhuber. 1997. Long Short-Term Memory. (1997).

[9] W. Hu, J. Gao, Y. Wang, O. Wu, and S. Maybank. 2014. Online Adaboost-Based Parameterized Methods for Dynamic Distributed Network Intrusion Detection. *IEEE Transactions on Cybernetics* 44, 1 (2014), 66–82.

[10] Neminath Hubballi and Vinoth Suryanarayanan. 2014. False Alarm Minimization Techniques in Signature-Based Intrusion Detection Systems: A Survey. *Computer Communications* 49 (08 2014), 1–17. https://doi.org/10.1016/j.comcom.2014.04.012

[11] R. U. Khan, X. Zhang, M. Alazab, and R. Kumar. 2019. An Improved Convolutional Neural Network Model for Intrusion Detection in Networks. In *2019 Cybersecurity and Cyberforensics Conference (CCC)*. 74–77.

[12] S. Kishorwagh, V. K. Pachghare, and S. R. Kolhe. September 2013. Survey on intrusion detection system using machine learning techniques. *International Journal of Computer Applications* 78, 16 (September 2013), 30–37. https://doi.org/10.5120/13608-1412

[13] Ron Kohavi. August 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *IJCAI'95: Proceedings of the 14th international joint conference on Artificial intelligence* 2 (August 1995), 1137–1143.

[14] F. Kuang, W. Xu, and S. Zhang. May 2014. A novel hybrid KPCA and SVM with GA model for intrusion detection. *Applied Soft Computing* 18 (May 2014), 178–184. https://doi.org/10.1016/j.asoc.2014.01.028

[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[16] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li. June 2014. A new intrusion detection system based on KNN classification algorithm in wireless sensor network. *Journal of Electrical and Computer Engineering* 2014 (June 2014).

[17] H. J. Liao, C. H. R. Lin, Y. C. Lin, and K. Y. Tung. 2013. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications* 36 (01 2013), 16–24. Issue 1. https://doi.org/10.1016/j.jnca.2012.09.004

[18] Mr.Rushi Longadge, Ms. Snehlata S. Dongre, and Dr. Latesh Malik. February 2013. Class Imbalance Problem in Data Mining: Review. *International Journal of Computer Science and Network (IJCSN)* 2 (February 2013). Issue 1.

[19] N. Moustafa and J. Slay. 2015. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*. 1–6.

[20] Z. Muda, W. Yassin, M. N. Sulaiman, and N. I. Udzir. 2011. Intrusion detection based on k-means clustering and OneR classification. In *2011 7th International Conference on Information Assurance and Security (IAS)*. 192–197.

[21] M. Panda, A. Abraham, S. Das, and M. R. Patra. 2011. Network intrusion detection system: A machine learning approach. *Intelligent Decision Technologies* 5, 4 (2011), 347–356. https://doi.org/10.3233/IDT-2011-0117

[22] M. Schuster and K. K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[23] T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li. 2020. BAT: Deep Learning Methods on Network Intrusion Detection Using NSL-KDD Dataset. *IEEE Access* 8 (2020), 29575–29585.

[24] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad. 2019. Survey on SDN based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications* 12 (2019), 493–501. https://doi.org/10.1007/s12083-017-0630-0

[25] A. K. Verma, P. Kaushik, and G. Shrivastava. 2019. A Network Intrusion Detection Approach Using Variant of Convolution Neural Network. In *2019 International Conference on Communication and Electronics Systems (ICCES)*. 409–416.

[26] R Vinayakumar, K.P. Soman, and Prabaharan Poornachandran. July-September 2017. Evaluation of Recurrent Neural Network and its Variants for Intrusion Detection System (IDS). *International Journal of Information System Modeling and Design* 8 (July-September 2017). Issue 3.

[27] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu. 2018. HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection. *IEEE Access* 6 (2018), 1792–1806.

[28] S. Yang. 2019. Research on Network Behavior Anomaly Analysis Based on Bidirectional LSTM. In *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*. 798–802.

[29] Chao Yin. 2011. An Improved BM Pattern Matching Algorithm in Intrusion Detection System. *Applied Mechanics and Materials* 148-149 (12 2011), 1145–1148. https://doi.org/10.4028/www.scientific.net/amm.148-149.1145

[30] J. Zhang, M. Zulkernine, and A. Haque. 2008. Random-Forests-Based Network Intrusion Detection Systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 5 (2008), 649–659.