Journal of Contemporary Issues in Business and Government Vol. 27, No. 06, 2021
https://cibgp.com/

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

An Efficient Network Intrusion Detection Model Combining CNN and BiLSTM

L K Suresh Kumar
University College of Engineering, Osmania University, India

Abstract

The technological advancement led to increase in the internet usage and created rooms for attackers to exploit our data. Hackers commonly conduct network attacks to alter, damage, or steal private data. Intrusion detection systems (IDS) are the best and most effective techniques when it comes to tackle these threats. An IDS is a software application or hardware device that monitors traffic to search for malicious activity or policy breaches. Intrusion detection is a major challenge for security experts in the cyber world. Traditional IDS failed to detect complex and unknown cyber-attacks. Many IDS models using machine learning (ML) methods have shown good performance in detecting attacks. However, their limitations in terms of data complexity give rise to DL methods. Recent work has shown that deep learning (DL) techniques are highly effective for assisting network intrusion detection systems (NIDS) in identifying malicious attacks on networks. This paper proposed a deep learning model that incorporates learning of spatial and temporal data features by combining the distinct strengths of a Convolutional Neural Network and a Bi-directional LSTM. The publicly available dataset NSL-KDD is used to train and test the model in this paper. The proposed model has a high accuracy rate of 99.22% and detection rate of 99.15%.

Keywords BiLSTM , CNN , Deep Learning , Intrusion Detection , LSTM , NIDS

1 Introduction

With the advancement in technology, the number of people connecting to the internet is increasing rapidly. As of April 2022, there were five billion internet users worldwide, which is 63 percent of the global population [1]. The internet is a great place to find information, but it's also full of dangers. These dangers include information theft, malicious attacks, viruses, scams, and hacking. Cyber security is a technique of protecting our systems and sensitive information from malicious attacks. Cybersecurity helps in ensuring that your data is not stolen or hacked.
A cyberattack is a process of attempting to steal data or gain unauthorized access to a computer or network. The most common cyberattacks include Man in the middle, DDoS (Distributed Denial of service), Phishing, Ransomware, Man in the Middle attack (MITM), SQL Injection, and DNS spoofing [2].
The biggest challenge faced by companies in Cyber Security is the implementation of effective Cybersecurity measures due to the ever-evolving nature of risks. It is as if there are more devices than people in this world today, and attackers are becoming more and more creative. The modern attackers are highly motivated to conduct the attacks and have enough resources, time, and money to support them in achieving the targeted goals. Attackers conduct the attacks in large numbers and in a well-organized and sophisticated manner. The attack causes a threat

to the confidentiality, integrity, or availability (CIA) of an information system. Some attacks are witnessed recently includes ransomware attack on the largest American oil pipeline company Colonial Pipeline, Pegasus spyware, etc.
Cybercrime costs the global economy about 1 trillion. The average cost of a cybersecurity attack has been increasing over time, according to IBM the average cost of a cybersecurity breach is $3,860,000. This is a 6.4% increase in their estimate for 2017 [3]. Fig.1 provides the number of cyberattacks during 2016 – 2020 time period [4].

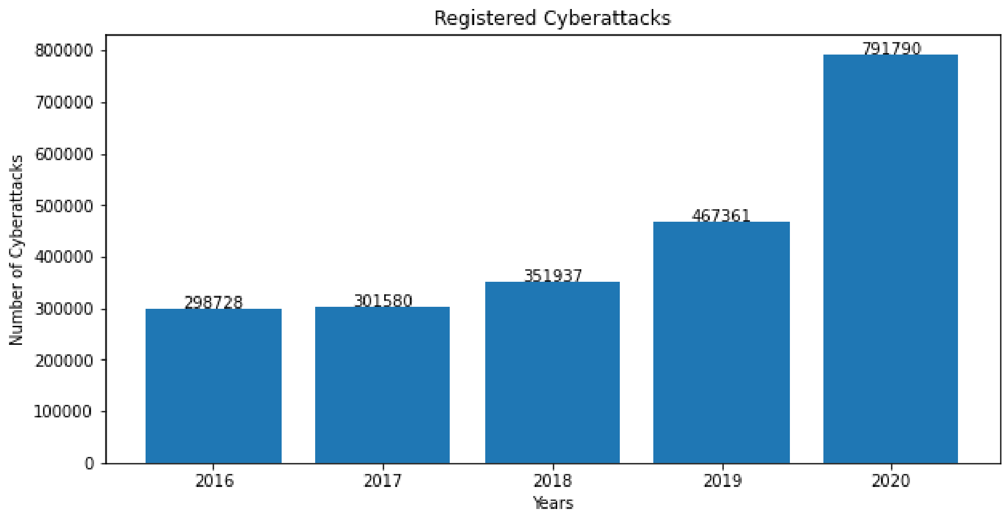


Fig. 1 Registered Cyberattacks per year

Organizations employs tools like firewall, antivirus software, and intrusion detection system (IDS) to ensure the security of the network and system within a cyberspace. Among these, network-based intrusion detection system (NIDS) is the attack detection mechanism that provides the desired security by constantly monitoring the network traffic for malicious and suspicious behaviour. Earlier network intrusion detection was conducted manually, where all the network activities are monitored, collected, and analysed with the help of system analysts and system administrators to find out the malicious activities. However, with the advent of technologies and increased users, the size and complexity of network flow also increased due to which manual intrusion detection became very hard or nearly impossible. To fulfill the requirements of an effective IDS, the researchers have explored the possibility of using machine learning (ML) and deep learning (DL) techniques.
The ML-based IDS depends heavily on feature engineering to learn useful information from the network traffic. While DL-based IDS do not rely on feature engineering and are good at automatically learning complex features from the raw data due to its deep structure. Over the last decade, various ML- and DL-based solutions were proposed by the researchers to make NIDS efficient in detecting malicious attacks. However, the massive increase in the network traffic and the resulting security threats has posed many challenges for the NIDS systems to detect malicious intrusions efficiently. The research on using the DL methods for NIDS is currently in its early stage and there is still an enormous room to explore this technology within NIDS to efficiently detect intruders within the network.

This paper is written with the motivation to build an efficient NIDS model using Deep Learning techniques which can detect attacks efficiently and provide security to our system.

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

This paper discusses a comprehensive literature review for understanding the IDS (Intrusion Detection System), with its types and Deep Learning techniques. Furthermore, based on the literature review, a framework to design NIDS (Network Intrusion Detection System) models has been proposed. At last, one NIDS model is presented as a proof of concept. The major contributions of the paper are as follows:

1. The review of IDS system with its types.
2. The review of feature optimization using Convolutional neural network (CNN).
3. The review of Deep learning (DL) model using BiLSTM for detecting intrusion with its importance to ascertain the efficacy and reliability.
4. Proposal of a framework based on the literature review to design an efficient and effective NIDS.
5. Simulation of a NIDS model based on the proposed framework as a proof of concept.
6. Evaluation of the developed NIDS model on a relevant benchmark networking dataset – NSL-KDD.

The rest of the paper is organized as follows: Section 2 provides an idea of IDS system and its types. Section 3 presents a literature survey for research works in NIDS. Section 4 presents Convolutional Neural Network model. Section 5 provides an understanding about Long Short Term Neural Network. Section 6 represents a proposed NIDS framework based on the survey. Section 7 represents the results and comparative analysis of the developed NIDS model based on the proposed framework. Finally, Section 8 concludes the paper.

2 Intrusion Detection System

Due to increase in the size of network and internet usage the modern attackers have enough resources, time, tools and techniques for performing attacks easily. The conventional methods like firewalls, encryption and anti-virus software packages adopted by organizations play a significant role in securing network infrastructure. Firewall acts as first line of Defence against network attacks. They monitor network traffic in order to prevent unauthorized access. Still, these methods provide the first level of defence and cannot completely protect the networks and systems from progressive attacks and malware. As a result, some intruders still manage to penetrate, resulting in a breach. Hence organisations use IDS (Intrusion detection systems). Intrusion is an unauthorised access to information within a computer or network. The IDS (Intrusion Detection System) analyzes the network packets, when an intrusion is detected an alarm is generated to drop the offending packets or terminate the connection [5]. Based on the data collection the IDS is classified into two types. Classification of IDS is provided in Fig. 2 .

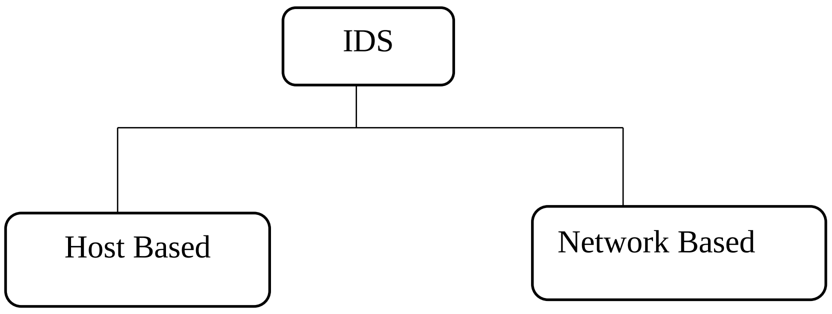


Fig.2 Classification of IDS

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

Host-based Intrusion detection system (HIDS) :
It is designed to monitor the traffic of a particular host such as server or PC. It takes a snapshot of existing system files and compares it with the previous snapshot for detecting rogue process, unauthorised access, software's suitability, DoS attack on the host. If an intrusion is detected an alert is sent to the administrator. Many researchers proposed HIDS in their works, such as [6,7].
Network-based Intrusion detection system (NIDS) :
It is designed to monitor, capture and analyse the network traffic. A sensor is used to examine the network traffic packets and their content as per the intrusion detection rules, once a malicious packet is detected it drops or blocks the packets, and terminates the malicious connections. Examples of network-based IDS are Cisco hardware sensors of 4200 series, Catalyst 6500 switch IDS module, etc.

Due to the constant change in network structure, the traditional IDS are facing challenges in detecting the attacks effectively. Hence researchers proposed various IDS systems using feature selection strategies and Machine learning methods [8].
The common Machine Learning algorithms are KNN, SVM, Decision Tree and ANN [9]. The ML methods have shown good performance in achieving high detection accuracy. Still, there are some limitations of ML methods like handling raw, unlabeled or high dimensional data. To handle this data complexity issues Deep Learning (DL) methods were introduced to increase the efficiency in detecting novel attacks and improve the performance of the model [10].

3 Related Work

NIDS ensures the good security level of a network from various attacks. Various tools, approaches, and methods based on machine learning and deep learning are used for detecting intrusion in a network.

Gao et al. [11] applied Deep Belief Network (DBN), Support Vector Machine (SVM), and Artificial neural network (ANN) on the KDD CUP 1999 for Big Data prediction in IDS and found the DBN performance was better with an accuracy of 93.49 % than compared to other two algorithms. The deep hierarchical model is a deep neural network classifier of a combination of multilayer unsupervised learning networks, which is called as Restricted Boltzmann Machine, and a supervised learning network, which is called as Back-propagation network.

Li et al. [12] applied autoencoder for feature selection and DBN as a classifier to detect intrusion as malicious code. Autoencoder deep learning method is used to reduce the dimensionality of data. This could convert complicated high-dimensional data into low dimensional codes with the nonlinear mapping, thereby reducing the dimensionality of data, extracting the main features of the data; then using DBN learning method to detect malicious code. DBN is composed of multilayer Restricted Boltzmann Machines (RBM, Restricted Boltzmann Machine) and a layer of BP neural network. In their experiment, they found DBN has successfully improved the accuracy by less time 92.1 % on the KDDCUP'99 dataset.

Lotfollahi et al. [13] proposed an approach for encrypted traffic classification with the help of an autoencoder combined with a CNN and a classification layer. Their approach differentiated between the virtual private network (VPN) and non-VPN traffic. The approach achieved an F1 score of 98 % on the ISCX VPN-nonVPN dataset, which consists of traffic in PCAP format files [14]. Deep packet with CNN as its classification model achieved recall of 0.98 in application identification task and 0.94 in traffic categorization task. Wang et al. [15] perform a similar task by using 1D CNN with more than 99.5 % of precision and recall.

Keserwani, P.K et al. [8] proposed an NIDS model using Genetic Algorithm for feature selection and Deep Neural Network for classification on UNSW-NB15 dataset. The searchability of GA and local search heuristics is combined to eliminate the un-relevant features. The selected features are fed to the deep neural network (DNN), which has been designed by selecting appropriate parameters such as number of neurons in each layer, dropout rate, batch size, number of epochs to produce high accuracy in prediction. The achieved accuracy and detection rate of the proposed model are 98.11% and 97.81% respectively.

Wang et al. [16] designed an intrusion detection algorithm and tested them in the CTU-13 dataset. In the preprocessing of the designed algorithm, the raw network data were converted into images, and then, the images are fed to a convolutional neural network (CNN). In the classification step, two approaches were followed: First, by using a 20-class classifier to identify traffic types as normal or malicious and achieve the accuracy of 99.17 %. In the second approach, a binary classifier was inputted to two CNNs for identifying traffic types as malicious or binary with an accuracy of 100 %.

Xiao et al. [17] proposed an efficient intrusion detection model on the KDDcup99 dataset with the help of PCA (principal component Analysis) and Autoencoder for dimensionality reduction and CNN is used for classification. The experimental results of the proposed CNN–IDS model shows an accuracy of 94.0% and also significantly reduces the classification time, which can satisfy the real-time requirements of the intrusion detection system. The main drawback is lower detection rates for the U2R and R2L classes compared to other attack classes.

Yu et al [18] proposed an IDS model based on novel DL idea of Few-shot Learning (FSL) [19]. The idea is to train using a small amount of balanced labeled data from the dataset. DNN and CNN are adopted as embedding functions in the model for extracting the essential feature and reducing the dimension. The proposed model using Few-Shot Learning (FSL), used less than 1% of NSL-KDD KDDTrain+ dataset for training, and achieved high accuracy of 92.34% for KDD-Test+ and 85.75% for KDD-Test-21.

Zhang et al [20] proposed a complex multilayer IDS model based on CNN and gcForest. They also proposed a novel P-Zigzag algorithm for converting the raw data into two-dimensional greyscale images. They used an improved CNN model (GoogLeNetNP) in a coarse grain layer for initial detection. Then in the fine-grained layer, gcForest (caXGBoost) is used to further classifies the abnormal classes into N-1 subclasses. They used a dataset by combining UNSW-NB15 and CIC-IDS2017 datasets. The experimental results show that the proposed model significantly improves the accuracy and detection rate compared to the single algorithms while reducing the FAR.

Shoayee Dlaim Alotaibi et al [21] proposed a DT-PCA-DNN intrusion detection model on NSL-KDD dataset and achieved an accuracy of 88.64% and detection rate of 84.56%. The model employs DT to do a preliminary screening of the preprocessed data to be detected before employing PCA as an input to perform a secondary judgment via DNN. The addition of DT causes a small increase in training time but a large boost in accuracy. Simultaneously, DT

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

prescreening reduces future DNN burden, which has a considerable effect on overall training pace.

Maimo´ et al. [22] proposed a method for anomaly detection and tested the method on the CTU dataset of botnet attacks [23]. They used DBN and RNN DL techniques as classifiers in sequence in their method. If the packet by the DBN was malicious, then the packet was inputted to RNN. The DBN with two hidden layers achieved precision, recall of 81.26 %, and 99.34 %, respectively, on the training dataset. Results demonstrate that the deep-learning artificial neural network model can accurately and efficiently identify botnets.

Kang and Kang [24] proposed a DL-based approach to detect intrusions in the in-vehicle network environment, where they used DBN with eleven layers DL method for classifier and achieved the accuracy and FAR of 97.8 % and 1.6 %, respectively. The DNN provides the probability of each class to discriminate normal and hacking packets, thus system can identify any malicious activity to the vehicle as a result. Features of in-vehicle network communication were collected from the controller area network (CAN) packets.

Diro and Chilamkurti [25] utilized the autoencoder with three hidden layers and achieved an accuracy of 99.2 % for binary classification and 98.27 % for multiclass classification. They proposed a DL classification algorithm for intrusion detection in the IoT network of fog computing and achieved. They tested the DL algorithm on the NSL-KDD dataset and achieved very good accuracy, DR, FAR 99.2 %, 99.27 %, 0.85 %, respectively.

Raman et al. [26] used the combination of hypergraph and Genetic Algorithm (GA) for feature selection and Support vector Machine (SVM) for classification in their proposed IDS. They tested their HG-GA-SVM model on the NSL-KDD dataset with the DR and FAR of 97.14 % and 0.83 %, respectively.

In [27], the authors have run 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 [27] have reported 64.8% accuracy from a GRU network and 67.5% from a LSTM Network.

In [28], 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.

4 Convolutional Neural Networks

Convolutional neural networks (CNNs) [29] have achieved excellent research results in computer vision [30], speech recognition, and natural language processing fields. For the IDS, they are used for the supervised feature extraction and classification purposes. It can learn features automatically and is better than other traditional feature selection algorithms.
It takes this name from mathematical linear operation between matrixes called convolution [31].

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

The Convolutional Neural Networks, which are also called as covnets, are neural networks which can share their parameters.
The CNN consists of an input layer, a convolutional layer, a pooling layer, a fully connected layer, and an output layer. A convolution neural network has multiple hidden layers that help in extracting information from an image. Every image is considered as a matrix of pixel values. The CNN model structure is presented in Fig. 3 .

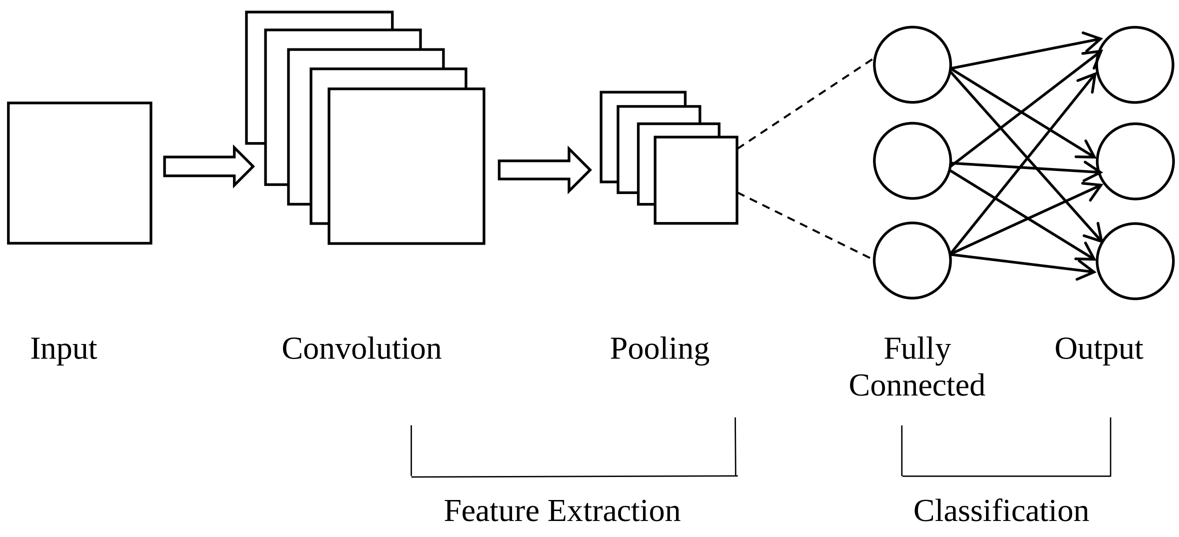
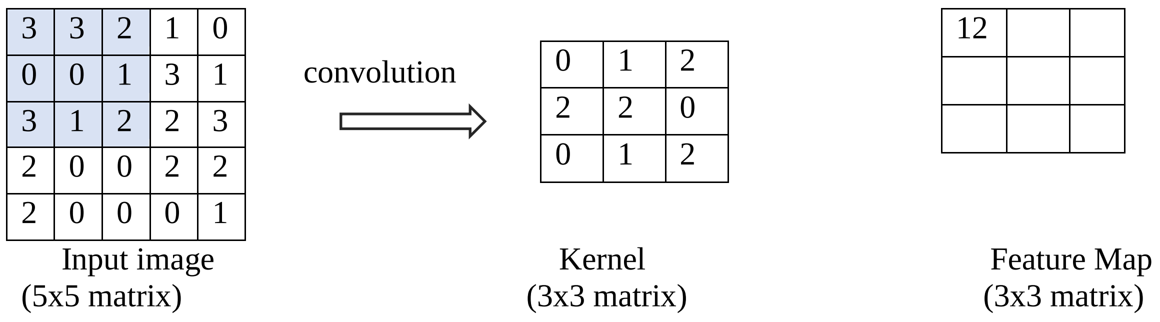


Fig. 3 CNN model structure

4.1 Convolution layer

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It consists of input data, a filter, and a feature map.
The feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution.
The filter is sub part of the image. It is a matrix that moves over the input image performing the dot product between the input pixels and the filter to get an output array. The resultant output array is called as feature map. Fig. 4 shows the convolution operation between input image and kernel.



3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

Input image
(5x5 matrix)

0	1	2
2	2	0
0	1	2

Kernel
(3x3 matrix)

12		

Feature Map
(3x3 matrix)

Fig. 4 Convolution Layer operation

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

Afterwards, the filter shifts by a stride, repeating the process until kernel has swept across the entire image. The weights in the kernel remain fixed as it moves across the image, which is also known as parameter sharing.
After each convolution operation, ReLU (Rectified Linear Unit) is applied by CNN to introduce non-linearity in the model. ReLU performs an element-wise operation and sets all the negative pixels to 0.
The convolution function [32] can be written as:

$$h_j = f(h_{j-1} \otimes w_j + b_j) \qquad (1)$$

where h_j is feature map of layer j,(j = 1, 2, . . . n)
b_j is bias of layer j
$\otimes$ is convolution function
f(x) is activation function (ReLU).

4.2 Pooling Layer

This layer is also called down sampling layer, it reduces the dimensionality of feature map (reducing the number of parameters in the input).
Through pooling we can reduce the size of feature map h_j and avoid over-fitting. It can be written as:

$$h_j = \text{pool}(h_{j-1}) \qquad (2)$$

We have 2 ways of down sampling i.e; Max pooling and Average pooling.
Max pooling: The pixel with the maximum value in feature map is sent to output array.
Average pooling: The average value of the pixels in receptive field is sent to output array.
Max pooling is depicted in Fig.5 .

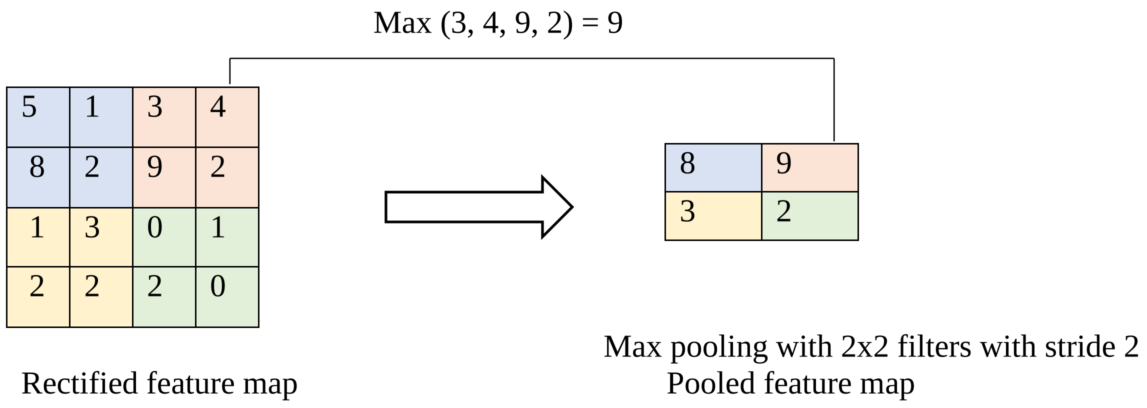


Fig. 5 Max Pooling

4.3 Fully Connected layers

Flattening is used to convert the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector as shown in Fig. 6 . The flattened matrix is fed as input to the fully connected layer to classify the image.

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

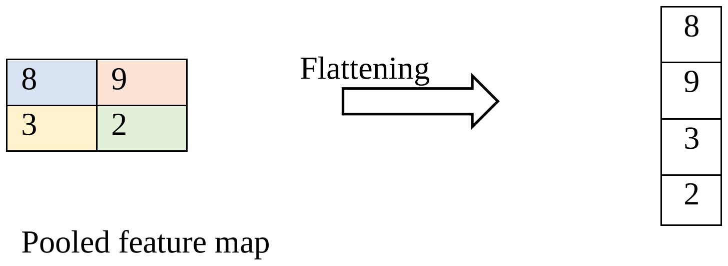


Fig.6 Flattening feature map

Fully connected layers connect every neuron in one layer to every neuron in another layer similar to a multiperceptron.

After several layers of convolution and pooling the output y_i vector is achieved through fully connected layer. Therefore, CNN is suitable for huge network data to extract the spatial features of network traffic data.

5 BiDirectional Long Short Term Memory

LSTM is special kind of Recurrent Neural Network that is capable of learning and allows information to persist. LSTM [33] is designed with a memory module that can determine when to keep memory and when to forget certain information.

A LSTM cell contains simple RNN cells, cell state (long term memory), forget gate, input gate, output gate and three logistic sigmoid gates and one tanh layer to limit the information passing through the cell. The output is usually in the range of 0-1 where ‘0’ means ‘reject all’ and ‘1’ means ‘include all’. The basic structure of an LSTM cell is depicted in Fig.7

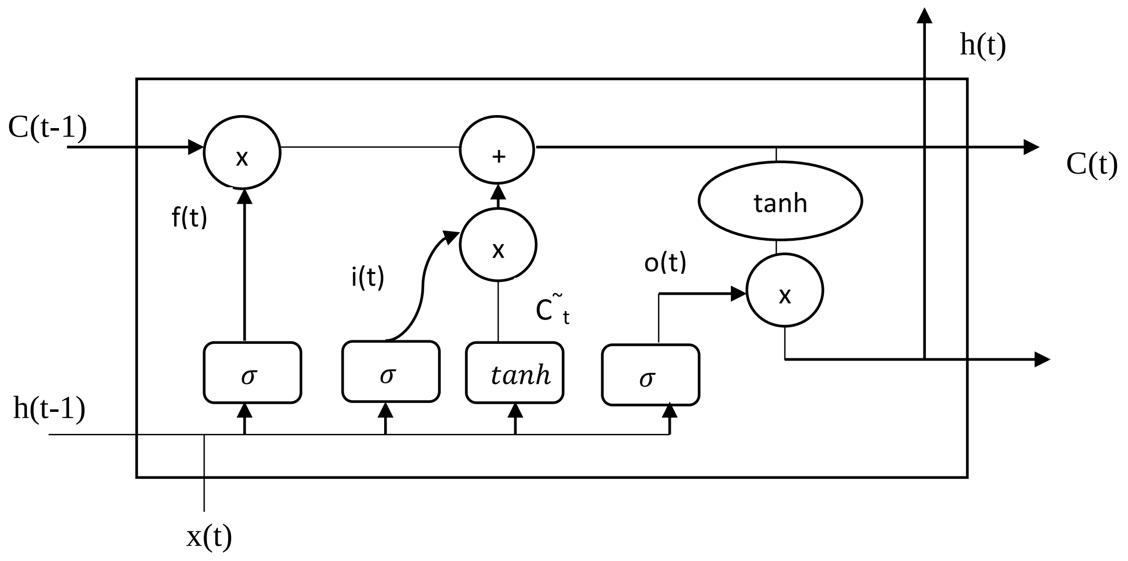


Fig. 7 Structure of a LSTM cell

Based on the structure of a LSTM cell [32] the calculation for a cell is given as:

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

At time t, the input gate is input according to the output result h_{t-1} of the cell at the previous moment. The input x_t at the current moment determines whether to update the current information into the cell through calculation. Here, W is the weight, and b is the bias of neurons and *t-1* indicates the previous moment.

$$i_t = \mathrm{sigmoid}(W_t \cdot [h_{t-1}, x_t] + b_t) \qquad (3)$$

Forget gate is based on the last moment hidden layer output h_{t-1} and the current time input is used to decide whether retain or discard the information.
Here f denotes values near forget gate.

$$f_t = \mathrm{sigmoid}(W_f \cdot [h_{t-1}, x_t] + b_f) \qquad (4)$$

The current candidate memory cell value is determined by the current input data x_t and the output result h_{t-1} of the LSTM hidden layer cell at the previous moment. In the current moment, the memory cell state value C_t is adjusted by the current candidate cell C_t and its own state C_{t-1}, input gate and forget gate. Character $*$ is the element-wise matrix multiplication.

$$C^{\sim}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \qquad (5)$$

$$C_t = f_t * C_{t-1} + i_t * C^{\sim} \qquad (6)$$

Calculate the output gate o_t , and the output is used to control the cell status value. The output of last cell is h_t , which can be expressed as

$$o_t = \mathrm{sigmoid}(W_o \cdot [h_{t-1}, x_t] + b_o) \qquad (7)$$

$$h_t = o_t * \tanh(C_t) \qquad (8)$$

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.
The BiLSTM [32] is composed of two LSTM networks, one forward LSTM and one backward LSTM. The forward LSTM hidden layer is responsible for the forward feature extraction and the backward one is responsible for backward feature extraction.
The state of BiLSTM at time t includes forward output and backward output.

$$h_t^{forward} = LSTM^{forward}(h_{t-1}, x_t, C_{t-1}) \qquad (9)$$

$$h_t^{backward} = LSTM^{backward}(h_{t-1}, x_t, C_{t-1}) \qquad (10)$$

$$H_t = [h_t^{forward}, h_t^{backward}] \qquad (11)$$

6 Proposed framework

The researchers have designed many IDS frameworks but not many of them addressed the issue of data imbalance. The framework should be designed in such a way that it ensures

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

security and detects even minority class attacks and handles the data imbalance issue and is adaptive in terms of organizational security needs.
An IDS model can be created using any combination of one feature selection method, one classification method and the model should satisfy the performance metrics such as accuracy, detection rate, false alarm rate. Researchers have also used ML models 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. This led to the adoption of deep learning approaches in order to resolve the lacking points of the above.

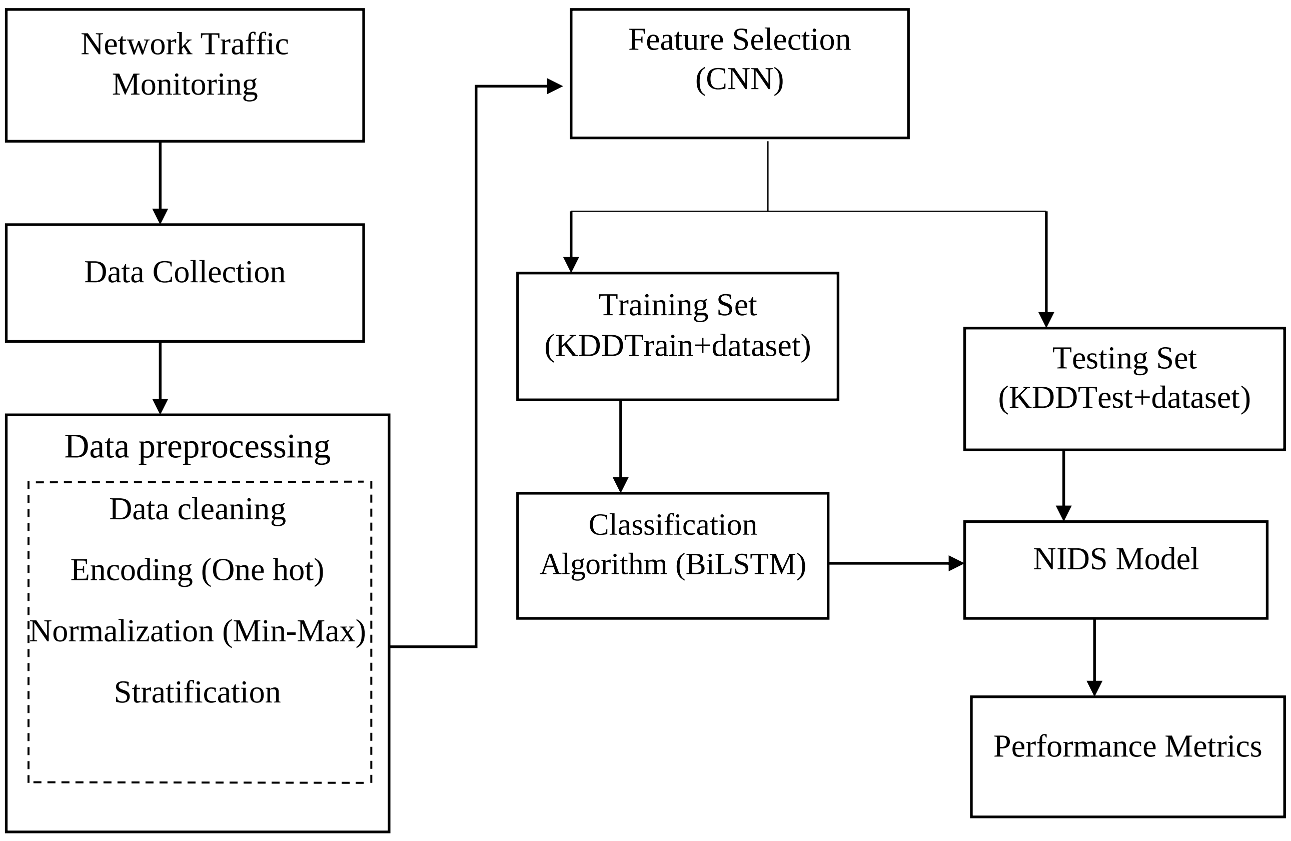


Fig. 8 Proposed framework for NIDS

Different modules presented in the framework are:

6.1 Dataset Creation

Dataset Creation includes three methods such as Monitoring network, Data collection and Feature extraction. The standard benchmark datasets available on internet are KDD-99, NSL-KDD, UNSW-NB15, CICIDS-2017, etc.; The proposed model uses NSL-KDD dataset which contains 22 attack types and 41 attributes. This is the revised and refined version of the KDD Cup'99 dataset by removing several of its integral issues.The label mainly contains normal data

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

and 4 types of attack data (DoS, Probe, U2R, R2L). It is made publicly available by The University of New Brunswick [34].
The datasets used in the proposed model are as below:

KDDTrain+.TXT:	The full NSL-KDD train set including attack-type labels and difficulty level in CSV format.
KDDTest+.TXT:	The full NSL-KDD test set including attack-type labels and difficulty level in CSV format.

The different attack categories [35] in NSL-KDD dataset are listed in Table 1.

Table 1: NSL-KDD Dataset Attack Categories

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

6.2 Preprocessing

In this module the created dataset is processed using cleansing, encoding and normalizing techniques to get quality results.

Data Cleansing :
The incorrect, incomplete or noisy data problems are handled by removing or updating them.

Encoding :
The data is encoded into required format using methods such as one hot, mean, label, ordinal, binary and frequency.

One hot Encoding :
There are categorical features in the NSL-KDD dataset that should be converted to numerical values for our deep learning model to produce accurate predictions. This process of converting categorical data variables into numerical values is called One hot encoding. As a result, in the pre-processing, these columns were converted into numerical values using the pandas python library's get-dummies function.

Normalization :
Normalization is the process of rescaling data into specific range in order to reduce redundancy and improve model training time.The paper employs Min-Max Normalization, which rescales the data range to [0,1].

$$x_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Stratified K-cross fold validation :
This technique is used for validation purpose and it requires less computation power.
Each fold in this technique is a good representation of the whole dataset.
Stratified K-cross fold technique splits the entire dataset into K sets and K-1 folds are used for training while Kth fold is used for validation. This is continued until all the folds are used to validate the model once.

6.3 Feature selection

There are many feature selection techniques, in our proposed model CNN (convolutional neural network) is used for the selection of relevant features.

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

The CNN network can use the hidden layers to learn the local features of the data layer by layer and extract the data features in the spatial dimension.

6.4 Classification

BiLSTM network has the characteristics of long-term preservation of information and can extract features in the time dimension [36]. Thus CNN-BiLSTM [32] hybrid NIDS model is used.
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 its 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.

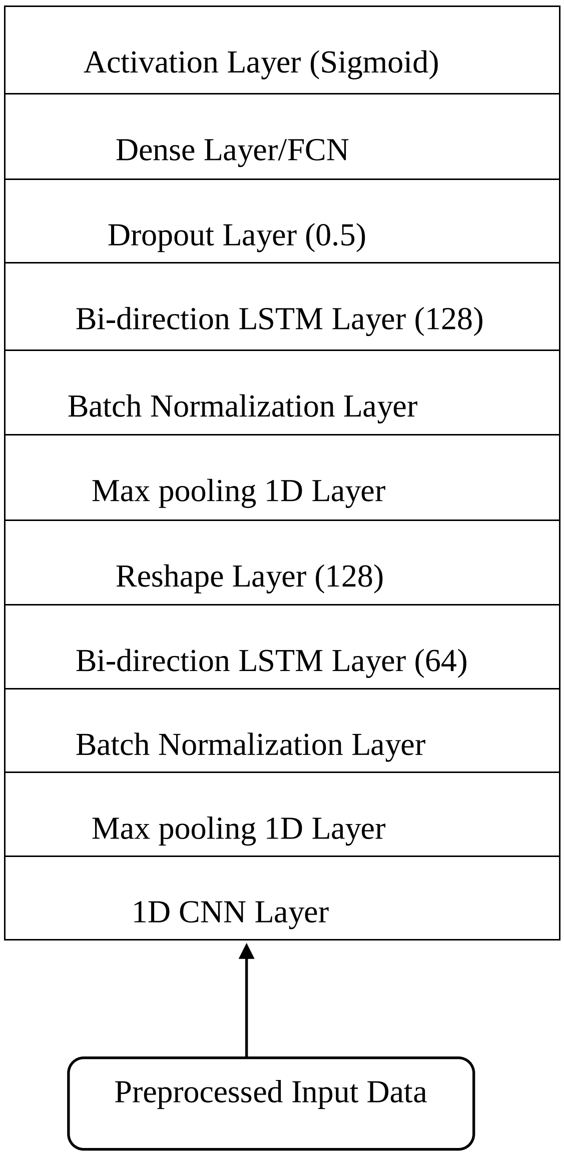

Activation Layer (Sigmoid)
Dense Layer/FCN
Dropout Layer (0.5)
Bi-direction LSTM Layer (128)
Batch Normalization Layer
Max pooling 1D Layer
Reshape Layer (128)
Bi-direction LSTM Layer (64)
Batch Normalization Layer
Max pooling 1D Layer
1D CNN Layer
Preprocessed Input Data

Fig.9 Block Diagram of model used

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

The 1-D CNN layer is followed by a Max Pooling 1D layer to recognise the relevant features resulting reduced training time and prevention from overfitting. After Max Pooling, comes Batch Normalization layer which enables normalization of parameters between intermediate layers to prevent slower training times. Reshape Layers after batch normalization layers reshape the output of the previous layer for the upcoming pair of Bi-LSTM layers.

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 Fully Connected Dense layer comes next which serves as an Output Layer followed by a Dropout Layer. The Dropout Layer is put in place to randomly drop units and connections from designed model, remove noisy data and prevent over Fitting. Even though the model uses Max Pooling in between every layer to prevent overfitting we use dropout layers. The reason behind this is that, generally, CNN and RNN used in combination have a higher probability of overfitting and perform poorly on the testing set.

6.5 Training Phase

The model is trained against KDDTrain+.TXT dataset which is 80% of the original dataset.

6.6 Testing Phase

The proposed model is tested against KDDTest+.TXT dataset which is 20% of the original dataset that is used to evaluated the performance.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_5 (Conv1D)            (None, 122, 64)           7872
_________________________________________________________________
max_pooling1d_9 (MaxPooling1 (None, 24, 64)            0
_________________________________________________________________
batch_normalization_9 (Batch (None, 24, 64)            256
_________________________________________________________________
bidirectional_9 (Bidirection (None, 128)               66048
_________________________________________________________________
reshape_5 (Reshape)          (None, 128, 1)            0
_________________________________________________________________
max_pooling1d_10 (MaxPooling (None, 25, 1)             0
_________________________________________________________________
batch_normalization_10 (Batc (None, 25, 1)             4
_________________________________________________________________
bidirectional_10 (Bidirectio (None, 256)               133120
_________________________________________________________________
dropout_5 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_5 (Dense)              (None, 5)                 1285
_________________________________________________________________
activation_5 (Activation)    (None, 5)                 0
=================================================================
Total params: 208,585
Trainable params: 208,455
Non-trainable params: 130
```

Fig.10 Designed CNN-BiLSTM Model

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

6.7 Evaluation Metrics

To evaluate the CNN-BiLSTM model, this paper selects three indicators: AC (accuracy), DR (detection rate), and FAR (false alarm rate). A confusion matrix shows the number of correct and incorrect predictions made by the model against the actual outcomes (target value) in the data. We first introduce the four parameters TP, FP, FN and TN, which are used to compute AC, DR and FAR.
TP (True Positive) is the number of attack samples classified into the attack class.
FP (False Positive) is the number of normal samples classified into the attack class.
FN (False Negative) is the number of attack samples classified into the normal class.
TN (True Negative) is the number of normal samples classified into the normal class.

Accuracy: It is equal to the correct predictions divided by total predictions.

$$AC = \frac{TP+TN}{TP+TN+FP+FN}$$

Recall (Sensitivity): It is the ratio of true positive prediction and actual positives. It is also kno wn as detection rate (DR).

$$DR = \frac{TP}{TP+FP}$$

False-Positive Rate (FPR) or False Alarm Rate (FAR): It is equal to the false positive predicti ons divided by actual negative values in the dataset.

$$FPR = \frac{FP}{TN+FN}$$

7. Results and Discussion

Experiments used the TensorFlow under windows as the backend, encoded with Keras and Python. The learning rate of the network model in this paper was set to 0.001. The weight inactivation rate of Dropout in the regularization method was set to 0.5, and the experiment iteration is 100 times, each batch_size was set to 128.
The performance of the proposed CNN-BiLSTM model is evaluated by calculating the performance metrics such as accuracy, FAR, precision, f1-score, recall. Table 2 shows the performance of proposed NIDS model with varying k values.

Table 2: CNN-BiLSTM performance metrics

K - Value	Accuracy	Detection Rate	False Positive Rate
2	0.985106	0.984877	0.378071
4	0.991220	0.990466	0.238356
6	0.993536	0.992910	0.177252
8	0.995529	0.994519	0.137021
10	0.995960	0.994788	0.130288
Average	0.992270	0.991512	0.212198

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

The proposed model provides an average accuracy of 99.22% for the NSL-KDD dataset in multiclass, with the best accuracy of 99.59% for k-value=10. The average Detection Rate is 99.15%, with k-value = 10 yielding the best result of 99.47%. The average FPR percent is 0.21, with k-value=10 yielding the best value of 0.13.
The maximum accuracy is 99.59% and the detection rate is 99.47% when k is 10, so as the number of folds increases, more samples of each attack/normal class are available for the model to train and thus the model can classify them better. Variation of accuracy and detection rate with k values is plotted in fig.11. As the k value increases the false positive rate is decreasing it can be observed from fig.12 .

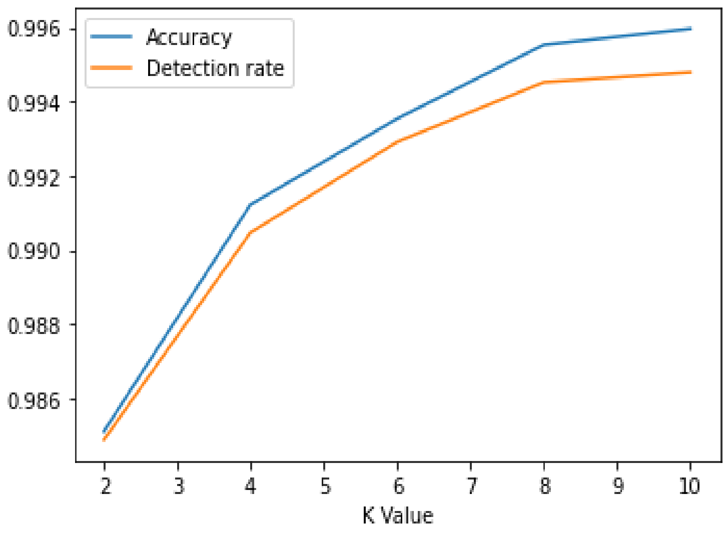


Fig. 11 Accuracy and Detection rate plot for NSL-KDD dataset

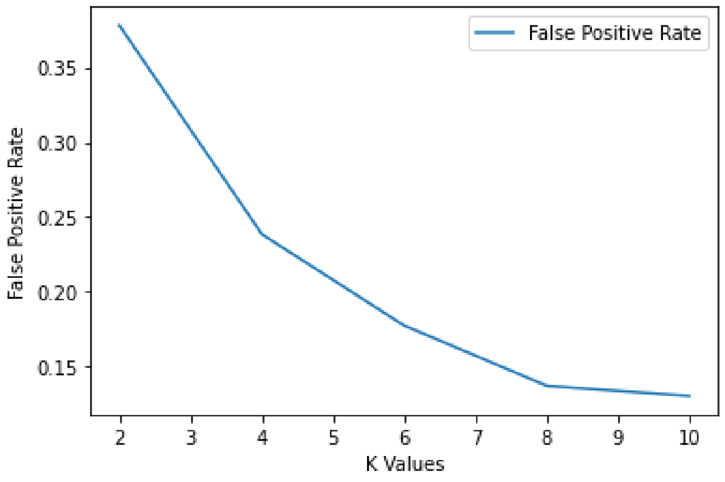


Fig. 12 False positive rate plot for NSL-KDD dataset

After evaluating the performance metrics of our proposed model, it is compared with existing models such as Few shot Learning, CNN, BiLSTM, Random Forest and BAT-MC Model on same NSL-KDD dataset to determine the best model which can give good results and detects attacks efficiently. The CNN-BiLSTM model has high accuracy and detection rate than compared with others.

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

Table 3: Comparative analysis of the proposed CNN-BiLSTM NIDS model with existing approaches

Model	**Accuracy (%)**	**DR (%)**
Proposed CNN BiLSTM Model	99.22	99.15
Few shot Learning	92.08	91.38
BAT-MC Model	83.54	84.29
Random Forest	75.09	70.26
CNN Model	82.30	85.24
BiLSTM Model	79.25	75.46

8. Conclusion

This paper proposes a model for analysing network traffic that takes into account a wide range of variables such as protocol type, service type, and so on. After training and testing on the NSL-KDD dataset, the proposed model yields promising prospective real-time usage for Intrusion Detection systems. The proposed model extracts feature automatically through repeated multi-level learning by taking advantage of the outstanding features of deep learning. The proposed CNN-BiLSTM NIDS model achieved accuracy and detection rate on the NSL-KDD dataset are 99.22% and 99.15%, respectively, achieving better performance than other approaches comparatively. Although the model would almost certainly need to be professionally monitored in a production environment, the risk of overfitting, misclassification, or error rate creep is significantly lower than that of many other machine learning models currently available. Furthermore, it necessitates fewer computing resources, less training time, and less specialized care during implementation and maintenance.

References

1. Global digital population as of April 2022
   https://www.statista.com/statistics/617136/digital-population-worldwide/
2. Types of Cyber Attacks
   https://www.fortinet.com/resources/cyberglossary/types-of-cyber-attacks
3. Understanding the cost of a cybersecurity attack: The losses organizations face | Packt Hub.
4. https://vpnusecase.com/statistics/cyber-attacks-per-year/
5. Liao H-J, Lin C-HR, Lin Y-C, Tung K-Y (2013) Intrusion detection system: a comprehensive review. J Netw Comput Appl 36(1):16–24
6. Hu J, Yu X, Qiu D, Chen H-H (2009) A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection. IEEE Netw23(1):42–47

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

7. Creech G, Hu J (2013) A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns. IEEE TransComput 63(4):807–819
8. Keserwani, Pankaj Kumar et al. “An effective NIDS framework based on a comprehensive survey of feature optimization and classification techniques.” *Neural Computing and Applications* (2021): 1-21. https://doi.org/10.1007/s00521021-06093-5
9. Dina, Ayesha S. and D. Manivannan. “Intrusion detection based on Machine Learning techniques in computer networks.” *Internet Things* 16 (2021):100462. https://doi.org/10.1016/j.iot.2021.100462.
10. Kocher, G., Kumar, G. Machine learning and deep learning methods for intrusion Detection systems: recent developments and challenges. *Soft Comput* **25,** 9731–9763 (2021). https://doi.org/10.1007/s00500-021-05893-0
11. Gao N, Gao L, Gao Q, Wang H (2014) An intrusion detection model based on deep belief networks. In: 2014 Second international conference on advanced cloud and big data.pp.247–252
12. Li Y, Ma R, Jiao R (2015) A hybrid malicious code detection method based on deep learning. Int J Secur Appl 9(5):205–216
13. Lotfollahi M, Siavoshani MJ, Zade RSH, Saberian M (2020) Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput 24(3):1999–2012
14. Draper-Gil G, Lashkari AH, Mamun MSI, Ghorbani AA (2016) Characterization of encrypted and VPN traffic using time-related. In: Proceedings of the 2nd international conference on information systems security and privacy (ICISSP), pp. 407–414
15. Wang W, Zhu M, Wang J, Zeng X, Yang Z (2017) End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: 2017 IEEE international conference on intelligence and security informatics (ISI), pp. 43–48
16. Wang W, Zhu M, Zeng X, Ye X, Sheng Y (2017) Malware traffic classification using convolutional neural network for representation learning. In: 2017 international conference on information networking (ICOIN), pp. 712–717
17. Y. Xiao, C. Xing, T. Zhang and Z. Zhao, "An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks," in IEEE Access,vol. 7, pp. 42210-42219, 2019, doi: 10.1109/ACCESS.2019.2904620.
18. Yu Y, Bian N. An intrusion detection method using few-shot learning. IEEEAccess.2020;8:49730-49740. https://doi.org/10.1109/ACCESS. 2020.2980136.
19. Wang Y, Yao Q, Kwok J, Ni LM. Generalizing from a few examples: a survey on few-shotlearning; 2019. arXiv: 1904.05046.
20. Zhang X, Chen J, Zhou Y, Han L, Lin J. A multiple-layer representation learning model for network-based attack detection. IEEE Access. 2019;7:91992-92008. https://doi.org/10.1109/ACCESS.2019.2927465.
21. Alotaibi, Shoayee et al. “Deep Neural Network-Based Intrusion Detection System through PCA.” *Mathematical Problems in Engineering* (2022): n. pag. https://doi.org/10.1155/2022/6488571
22. Maimo´ LF, Go´mez A´ LP, Clemente FJG, Pe´rez MG, Pe´rez GM (2018) A self-adaptive deep learning-based system for anomaly detection in 5g networks. IEEE Access 6:7700–7712
23. Garcia S, Grill M, Stiborek J, Zunino A (2014) An empirical comparison of botnet detection methods. Comput Secur 45:100–123
24. Kang M-J, Kang J-W (2016) Intrusion detection system using deep neural network for

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

in-vehicle network security. PloS one 11(6):e0155781

25. Abeshu A, Chilamkurti N (2018) Deep learning: the frontier for distributed attack detection in fog-to-things computing. IEEE Commun Mag 56(2):169–175
26. Raman MG, Somu N, Kirthivasan K, Liscano R, Sriram VS (2017) An efficient intrusion detection system based on hypergraph-genetic algorithm for parameter optimization and feature selection in support vector machine. Knowledge-Based Syst 134:1–12
27. R. Vinayakumar, K. Soman, and P. Poornachandran, "Evaluation of recurrent neural network and its variants for intrusion detection system (ids),"International Journal of Information System Modeling and Design, vol. 8,July-September 2017.
28. R. U. Khan, X. Zhang, M. Alazab, and R. Kumar, "An improved convolutional neural network model for intrusion detection in networks," in 2019 Cybersecurity and Cyberforensics Conference (CCC), 2019, pp. 74–77.
29. A. Krizhevsky, I. Sutskever, and G. E. Hinton, ''Imagenet classification with deep convolutional neural networks,'' Commun. ACM, vol. 60, no. 6, pp. 84–90, 2017.
30. Lawrence S, Giles CL, Tsoi AC, Back AD. Face recognition: a convolutional neural-network approach. IEEE Trans Neural Netw. 1997;8(1):98-113. https://doi.org/10.1109/72.554195.s
31. S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6,doi:10.1109/ICEngTechnol.2017.8308186.
32. K. Jiang, W. Wang, A. Wang and H. Wu, "Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network," in IEEE Access, vol. 8, pp. 32464-32476,2020, doi: 10.1109/ACCESS.2020.2973730.
33. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735-1780. https://doi.org/10.1162/neco.1997.9.8. 1735.
34. NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB. https://www.unb.ca/cic/datasets/nsl.html.
35. Sinha, J., & Manollas, M. (2020). Efficient Deep CNN-BiLSTM Model for Network Intrusion Detection. *Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Pattern Recognition*.
36. Gao, Jing. (2022). Network Intrusion Detection Method Combining CNN and BiLSTM in Cloud Computing Environment. Computational intelligence and neuroscience. 2022. 7272479. 10.1155/2022/7272479.
37. Kishor Kumar Reddy C, Anisha P R, Shastry R, Ramana Murthy B V, "Comparative Study on Internet of Things: Enablers and Constraints", Advances in Intelligent Systems and Computing, 2021
38. Kishor Kumar Reddy C, Anisha P R, Apoorva K, "Early Prediction of Pneumonia using Convolutional Neural Network and X-Ray Images", Smart Innovation, Systems and Technologies, 2021
39. R Madana Mohana, Kishor Kumar Reddy C and Anisha P R, "A Study and Early Identification of Leaf Diseases in Plants using Convolutional Neural Network", Springer 4th Int Conference on Smart Computing and Informatics, 2020, India

P-ISSN: 2204-1990; E-ISSN: 1323-6903
DOI: 10.47750/cibg.2021.27.06.140

40. Anisha P R, C Kishor Kumar Reddy and Nuzhat Yasmeen, “Predicting the Energy Output of Wind Turbine Based on Weather Condition”, Springer 4th Int Conference on Smart Computing and Informatics, 2020, India

41. Kishor Kumar Reddy C, Apoorva K, Anisha P R, “Early Prediction of Pneumonia using Convolutional Neural Network and X-Ray Images”, Springer 4th Int Conference on Smart Computing and Informatics, 2020, India

42. Viswanatha Reddy, Dr. Elango NM and Dr. C Kishor Kumar Reddy and Anisha P R, “Prediction of Diabetes using Internet of Things (IOT) and Decision Trees: SLDPS, Springer FICTA, January 2020, India.