I. J. Computer Network and Information Security, 2024, 2, 14-27
Published Online on April 8, 2024 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2024.02.02

Integrated Spatial and Temporal Features Based Network Intrusion Detection System Using SMOTE Sampling

Shrinivas A. Khedkar*
Veermata Jijabai Technological Institute, Mumbai, 400019, India
E-mail: sakhedkar@ce.vjti.ac.in
ORCID iD: https://orcid.org/0000-0002-1967-4799
*Corresponding Author

Madhav Chandane
Veermata Jijabai Technological Institute, Mumbai, 400019, India
E-mail: mmchandane@it.vjti.ac.in
ORCID iD: https://orcid.org/0000-0002-2872-4647

Rasika Gawande
Veermata Jijabai Technological Institute, Mumbai, 400019, India
E-mail: rrgawande_b18@ce.vjti.ac.in
ORCID iD: https://orcid.org/0000-0002-3583-2865

Received: 02 September 2022; Revised: 01 November 2022; Accepted: 29 December 2022; Published: 08 April 2024

Abstract: With attackers discovering more inventive ways to take advantage of network weaknesses, the pace of attacks has drastically increased in recent years. As a result, network security has never been more important, and many network intrusion detection systems (NIDS) rely on old, out-of-date attack signatures. This necessitates the deployment of reliable and modern Network Intrusion Detection Systems that are educated on the most recent data and employ deep learning techniques to detect malicious activities. However, it has been found that the most recent datasets readily available contain a large quantity of benign data, enabling conventional deep learning systems to train on the imbalance data. A high false detection rate result from this. To overcome the aforementioned issues, we suggest a Synthetic Minority Over-Sampling Technique (SMOTE) integrated convolution neural network and bi-directional long short-term memory SCNN-BIDLSTM solution for creating intrusion detection systems. By employing the SMOTE, which integrates a convolution neural network to extract spatial features and a bi-directional long short-term memory to extract temporal information; difficulties are reduced by increasing the minority samples in our dataset. In order to train and evaluate our model, we used open benchmark datasets as CIC-IDS2017, NSL-KDD, and UNSW-NB15 and compared the results with other state of the art models.

Index Terms: Network Intrusion Detection, CNN, BIDLSTM, SMOTE, CICIDS2017, Deep Learning.

1. Introduction

The number of people, businesses, and institutions relying on technology has grown as a result of recent technological advancements, and practically everyone now has access to the internet and a smartphone. As a result, information storage and transmission through networks have been sped up. As the number of system attacks rises quickly and attackers discover new ways to compromise the system, increased reliance on online services has also prompted many concerns about the security of the user. Numerous researchers have experimented with different machine learning methods. However, time-series data from network traffic data hasn't gotten much attention [1]. Deep Learning techniques, however, outperform machine learning because they can extract complicated information and carry out more computationally demanding tasks over various layers. Due to its ability to operate massive amount of data and perform calculations quickly and affordably, deep learning is highly scalable. Therefore, as fresh, larger datasets are created, the model can be trained. One of the biggest challenges in developing a reliable intrusion detection

system is this. When using deep learning, there are restrictions when experimenting with unbalanced data sets. Although the performance of the minority class is what the researchers are most worried about, the model will perform poorly in this population. As a result, we have included a sampling method known as the Synthetic Minority Oversampling Technique (SMOTE).

There is a lot of room for advancement in datasets like UNSW-NB15, CIC-IDS2017 and CIC-IDS2018, while much research is done on datasets like NSL-KDD and KDD Cup '99. Since these datasets are more than 20 years old, when subjected to contemporary attacks. The largest problem might be a lack of readily available datasets. In this paper, we suggest a CNN Bi-directional LSTM (BIDLSTM) combined with the SMOTE oversampling method. BIDLSTM reverses the information flow and adds an additional layer of LSTMs that comprehend context more effectively than traditional LSTMs.

The following are some important contributions that this study makes:

- We propose an integrated deep learning approach based on CNN-BIDLSTM a data feature extraction technique and generate synthetic samples and balance the dataset in order to address the imbalance of small attack samples.
- The use of CNN-BIDLSTM model accurately extracts the data's features and improves the detection rate compared to state-of-the art techniques.
- We may deduce that almost 60% of the experiments have been performed on NSL KDD and KDD Cup '99, although these do not cover recent attacks, based on an analysis of how public datasets have been used by researchers. Experiments on the CIC-IDS2017, NSL-KDD, and UNSW-NB15 validate our IDS model.
- Proposed model is consistent when evaluated on different metric as accuracy, sensitivity, specificity and F1 score to validate the performance.

2. Related Works

In the field of cyber security, a great deal of study has been done on intrusion detection systems and developing an IDS that can function successfully in a real-time context. In the past, intrusion detection systems based on signatures were more common. Deep learning and machine learning techniques have recently contributed significantly to intrusion detection systems. Researchers have created a number of algorithms and hybrid strategies to address this problem. We'll discuss a few strategies that are pertinent to our research.

The UNSW-NB15 dataset's pre-processing is enhanced in the study by Aleesa et al., [1] in order to be handled by deep learning models for the purpose of spotting anomalous patterns. The accuracy of the proposed models for binary classification was 99.26% for ANN, 99.22% for DNN, and 85.42% for RNN-LSTM. To identify network intrusion, Shende et al. (2020) [2] used a deep learning LSTM system. The accuracy of the LSTM model for binary and multi-class classification, respectively, was 99.2% and 96.9% using the NSL-KDD dataset for training and testing. KDDtrain+ and KDDTest+ files were combined in a 60:40 to train and test the model. For intrusion detection, Chuanlong Yin et al. (2017) [3] employed a recurrent neural networks method (RNN-IDS). Different neuronal counts and learning rates are used in the model's performance. The paper contrasts the model with ML approaches previously developed by researchers, including J48, support vector machine, random forest, and artificial neural network using the benchmark data set. The author proved that RNN IDS outperforms traditional machine learning classification methods in both binary and multiclass classification through experiment. The authors of [4] proposed a deep learning approach with modified ReLU activation function on CNN for detection of fault of rotating machineries. The proposed model has high convergence speed with accepted accuracy. The modified ReLU function is useful when the input has negative value. The multiplication factor in activation function has to be greater than zero if the multiplication factor is negative then output of activation function may not acceptable with negative sample.

Convolutional Neural Network and Long Short-Term Memory (LSTM) were combined in Mostofa Ahsan et. al.(2020) [5] hybrid technique to enhance intrusion detection. Despite the algorithm's 99.70% accuracy on the NSL KDD standard dataset, a significant high false positive rate for the Probe attack was found. Pengfei Sun et al. (2020) [6] developed a system that employs a hybrid network of CNN and LSTM's. A category weight optimization approach is used to reduce the impact of an unbalanced amount in order to increase model resilience. The model was trained exclusively on attacks like FTP-Patator, SSH-Patator, DoS, Infiltration, Heartbleed, and Portscan, leaving behind files for Bot, DDoS, and Web Attacks. Despite achieving a multi-class classification accuracy of 98.67 percent and an accuracy of each attack type of above 99.50 percent for the CIC-IDS 2017 dataset, however, these attacks were the only ones used in the model's training.

The problem of low detection accuracy and selection of appropriate features in intrusion traffic is handled using BAT-MC model [7]. The proposed model based on attention mechanism monitors the features from network flow selected using BLSTM. The model uses multiple convolution layers for local feature extraction. The proposed model is promising in terms of accuracy when compared with state of the art models. The Imrana et al. [8] identifies challenges faced by existing intrusion detection systems and problems in detecting User-to-Root (U2R) and Remote-to-Local (R2L) attacks. Therefore, a bidirectional LSTM based model is trained on NSL-KDD dataset and through experimentation shows effectiveness of proposed model compared to conventional LSTM. The proposed model shows reduced false

alarm rate with higher accuracy for U2R and R2L attacks. The authors recommended developing integrated system.

Bidirectional Long Short-Term Memory architecture and the UNSW-NB15 data set were used in SU Yang's (2019) [9] training and testing. The study found that, in terms of detection rate, the bidirectional LSTM network beats the conventional LSTM network. The experimental findings showed that the model is accurate in determining whether a group is normal or abnormal, but it is ineffective in identifying the sort of assault. To improve the detection system's capability, Aichuan Li et al. (2022) [10] created a CNN-based architecture that integrates with the BiLSTM network. The suggested CNN-BiLSTM model provides the best accuracy and performance, with a 97.1 percent detection rate, according to the testing results on the NSL-KDD data set. The problem of parameter explosion is addressed by the suggested fix. Praanna et al [11] proposed a CNN-LSTM IDS model for high dimensional data in which integrated folding and grouping approach derives the feature relationship. The KDD99 dataset used to compare the accuracy of proposed approach with Support Vector Machine, Deep Belief Networks and CNN*.

The survey by Hindy [12] have worked on available intrusion detection dataset with their problems and increasing threats. The research categorized threats based on threat sources, open system interconnection layer and active-passive modes of threats. The survey by Hongyu Liu et al. (2019) [13] mentions the shortcomings of intrusion detection systems in real-world circumstances. It stipulates that effective intrusion detection systems need to be very accurate at detecting intrusions in addition to being highly interpretable and runtime efficient. Since unsupervised learning can detect even when datasets are few and deep learning's fitting ability is superior, these two techniques are currently dominating new IDS investigations, the report claims. The Ferrag et al.[14] provides analysis of discriminative and unsupervised deep learning techniques in binary classification and multiclass classification. The CSE-CIC-IDS-2018 dataset used for analysis is recent benchmark dataset on key performance parameters. A one-class SVM and autoencoder outlier detection are being compared in the study H. Hindy et al. (2020) [15]. An unsupervised outlier-based ML technique such as one-class SVM is used as state-of-the art model. The Autoencoder model produced a respectable accuracy of 95.19 percent for the 0.15 threshold and an accuracy of 90.17 percent on the KDDTest+ dataset. The highly correlated features are extracted by making use of combined data analytics and statistical approach [16]. The proposed Autoencoder based and LSTM based approach is tested on NSL-KDD benchmark dataset. The AE_{50} outclassed all other state of the art models with accuracy of 84.21% and 87% on binary classification and multi-class classification respectively. Kim et al. [17] proposes a novel approach of generating synthetic malwares and separates them from actual malwares using transferred DCGAN. Using deep autoencoder malware features are extracted produces data and stabilizes GAN generator training. The tDCGAN based model outstands other machine learning based models with 95.74% average accuracy.

3. Deep Learning Intrusion Detection Model

This part explains the pre-processing of the dataset, explains the proposed model, and goes over the training and evaluation procedures.

By enabling the system to automatically comprehend and enhance by identifying patterns in the database without human interaction, machine learning attempts to create computer programs that can access data. A subset of machine learning known as "Deep Learning" makes use of an artificial neural network with several inputs, outputs, and hidden layers.

CNN has the convolutional layer helps to reduce the noise in data, the pooling layer to filter best features among the set and the fully connected layer. The convolution layer extracts features and pooling layer extracts the relationship between features at different places, producing reliable results and minimizing the overfitting issue. By serializing TCP/IP packets, 1D CNN is able to learn features of large network traffic. By increasing the number of convolution kernels, we use CNNs to learn spatial characteristics of traffic data, enabling both coarse- and fine-grained (at the network's end) learning. The CNN solves the problem of spatial feature extraction from traffic data however it fails to determine correlation information from long term sequential traffic data.

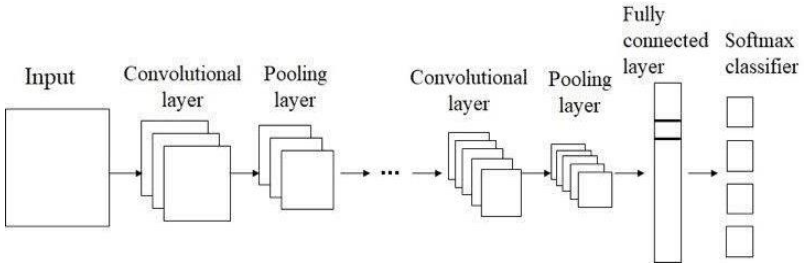

Fig.1. CNN architecture

A hidden state in a recurrent neural network (RNN) stores historical data by executing numerous loops. As seen in the figure 2, Long Short-Term Memory (LSTM) is an RNN with features called "gates" that regulate the flow of information through the device. They deal with RNN's vanishing gradient issue.

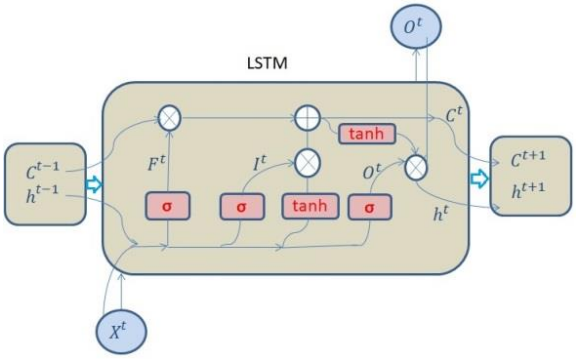

Fig.2. LSTM structure

Information from the past and the future can both be stored in the bidirectional long short-term memory (BIDLSTM), which combines two independent LSTMs that operate in the opposite way and can store information from the future. Concatenation (default) merge mode is used to combine the outputs of the two bidirectional LSTMs before moving on to the following layer.

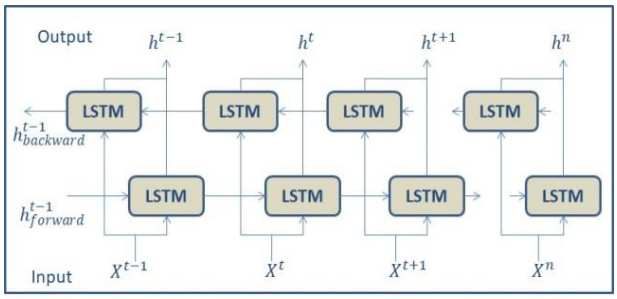

Fig.3. Bi directional LSTM structure

Synthetic Minority Oversampling Technique, or SMOTE for short, is a method for handling issues that can occur when employing an unbalanced data collection. SMOTE has the advantage of preventing the production of duplicate data points. Instead, it alters real samples to produce synthetic samples. The equation (1) is used to create a fresh example for each case:

$$z' = z + rand(0,1) * |z - z_k| \quad (1)$$

Where, z' is newly generated sample
z is original minority sample
z_k is randomly selected k sample

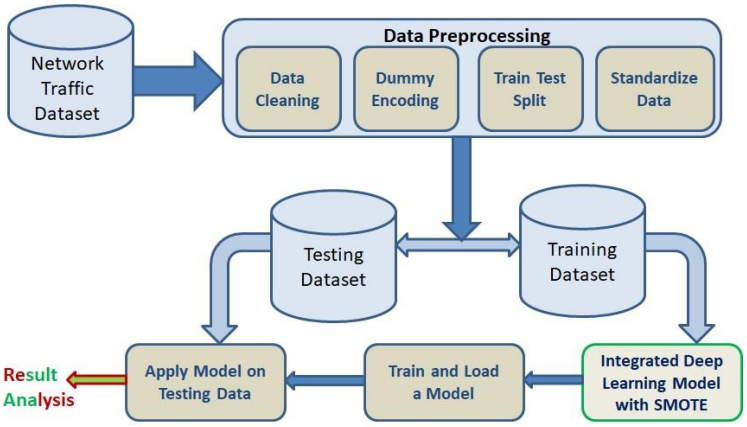

Fig.4. Deep learning intrusion detection model structure

3.1. Data Preprocessing

Records of network traffic are taken from the actual network traffic, hence there may be erroneous information (marked as "NaN" in the record). Cleaning up this erroneous data is the first stage in the preparation procedure. The

network dataset's features are divided into categorical and numerical categories. When encoding categorical features, we employ dummy encoding. With this approach of categorical data encoding, category variables are changed into dummy variables. N binary variables are utilized in N categories of one variable for one-hot encoding. In dummy encoding, N labels or categories are represented by N-1 features. Two distinct data frames are created by separating the features and labels data. Later, the stratified train-test split is carried out taking into account the label column. The training and test datasets are guaranteed to have the same proportion of target functionality as the original dataset thanks to stratified sampling. The data min-max scaler is used to standardize the data for numeric attributes. Each feature is scaled and transformed separately by this estimator to fit within the training set's defined range. Algorithm 1 explains the data pre-processing algorithm.

Algorithm1: Data Preprocessing

Load Dataset$\{NSL-KDD, CICIDS2017, UNSW-NB15\}$
Apply data cleaning
for $\forall$*Features in sample X do*
    if $x_{feature} = categorical\ then$
        apply one hot feature encoding
    endif
endfor
Extract Feature(X)
Extract Label(X)
Apply Stratified split on processed dataset
for *Numerical Features* ***do***
    Apply $min-max.normalization$
    $X_{trainscaled} = Z' * (max - min) + min$

$$X_{testscaled} = Z' * (max - min) + min$$

endfor

3.2. Integrated Spatial and Temporal Feature Extraction

We have considered the inefficiency of using a convolutional network alone to extract the temporal information of features from traffic data. CNN and the BIDLSTM are utilized to extract the spatial and temporal information, respectively, from the network traffic dataset. Along with extracting important features, it is necessary to delete duplicate feature information from the network traffic data. The intrusion detection structure is built into the preprocessing, SMOTE integrated CNN-BIDLSTM training, and model classification on testing data, as shown in Figure 4. An integrated spatial-temporal feature extraction consists of a feature input layer, a convolution layer, a pooling layer, a BIDLS layer, a fully connected layer, and output layers. The output data dimensionality is changed before the first BIDLSTM layer and after batch normalization is implemented.

By using a convolution layer and a pooling layer, the CNN model recovers local spatial data such an IP address or connection port number. The Pooling layer extracts the relationship between features at different places, producing reliable results and minimizing the overfitting issue. The "ReLU" activation function is used in the convolution layer. For all neurons that determine the number of weights, the convolution kernel size is the same. Convolution for the 1DCNN can be expressed as:

$$X_k = Conv1D(W_k \otimes a_{k-1}) + b_k \tag{2}$$

$$Y_k = f(X_k) \tag{3}$$

Where $X_k$is defined as input of k^{th}neuron, $W_k$is kernel from k^{th}neuron, a_{k-1} is the output of the previous kernel, b is b bias, and Y_k is the intermediate output.

Based on feature statistics, the pooling layer filters data and merges nearby feature points. The pooling layer decreases the problem of overfitting and shrinks the size of the feature.

$$Y_i = pool(Y_{i-1}) \tag{4}$$

The set of spatial features from the convolution layer and pooling layer of the 1DCNN are the output, designated as Y_i.

To successfully use for time series traffic data, we must now extract the features of reasonably large intervals. The sequence of network traffic dataset's gradient expansion and gradient disappearance issues are addressed by LSTM. However, LSTM structure can be significantly impacted by bidirectional sequential reading of network data. Memory cells, input gates, output gates, and forget gates make up the fundamental LSTM structure.

Initially, at time t, input X^t, weight W, the output of previous cell state $h^{t-1}$and bias of neuron b in LSTM forget

gate decides whether the information has to be discarded or retained.

$$F^t = sigmoid(W_F \cdot [h^{t-1}, X^t] + b_F) \quad (5)$$

Next, the inputX^t at the current cell state and output $h^{t-1}$of previous cell state determines new information based on calculation into the current state.

$$I^t = sigmoid(W_I \cdot [h^{t-1}, X^t] + b_I) \quad (6)$$

$$\mathbb{C}^t = tanh(W_C \cdot [h^{t-1}, X^t] + b_C) \quad (7)$$

$$C^t = F^t * C^{t-1} + I^t * \mathbb{C}^t \quad (8)$$

Finally, the output of cell state is determined by output gateO^t. The cell state output of last cell $h^t$is the result of output gate O^t multiplied by processed cell state C^tthrough $tanh$.

$$O^t = sigmoid(W_O \cdot [h^{t-1}, X^t] + b_O) \quad (9)$$

$$h^t = O^t * \tanh(C^t) \quad (10)$$

The states of bidirectional LSTM at any time t are

$$H^t = [h^t_{forward}, h^t_{backward}] \quad (11)$$

$$h^t_{forward} = h^t_{backward} = LSTM(h^{t-1}, X^t, C^{c-1}) \quad (12)$$

The result of integrated spatial and temporal features extraction is used to train the model for network intrusion detection.

3.3. Model Training

To solve the problem of the large false-negative rate, we propose a SMOTE integrated CNN-BiLSTM method. By producing "synthetic" instances, SMOTE oversamples the minority class, resolving the issue of imbalanced data and enabling the model to train on both types of data. The parameter in equation 2 must be calculated in order to identify a minority class. The classes with c smaller than t are regarded as minority classes when we compare the parameter with the threshold t. The threshold for the CIC-IDS 2017 is 0.001, whereas the threshold for the NSL-KDD is 0.1.

$$\beta^C = \frac{Q^c}{Q_{benign}} \quad (13)$$

$Where, Q^c$ $represents$ $attack$ $class$ $sample$ $count$

$Q_{benign}$$represents$ $legitimate$ $sample$ $count$

To learn spatial feature hierarchies, one uses CNN. Convolution layers, pooling layers, and fully connected layers are only a few of the building elements used in the creation of CNN. BIDLSTM has the ability to extract time-level data features and store context history information for a long time. So, to balance the benign and attack classes and ultimately create a deep hierarchical network model, we integrated SMOTE with CNN-BIDLSTM in this research. In algorithm 2, the model algorithm is described.

3.4. Performance Metrics

A method for intrusion detection must have a high true rate and a low false rate in order to be effective. The effectiveness of our suggested strategy is evaluated using the following metrics.

A. Mean Squared Error

How closely a fitted line resembles the data points are expressed in terms of the Mean Squared Error (MSE). Square the vertical distance between the data point and the corresponding y value on the curve fit to get the accuracy.

$$MSE = \left(\frac{1}{n}\right) * \sum(\text{actual} - \text{forecast})^2 \quad (14)$$

where n denotes the number of objects, Actual Equals original or observed y-value, Forecast Means regression y-value.

To evaluate the model performance we adopted accuracy, precision, recall and F1-score as key parameters.

Algorithm2: Proposed Training model

Input: *Imbalanced data* $(D), Q^C$ *attack class having* C
$C = 1,2,3, \dots, c;$ *threshold* $t = \{0.1, 0.001\}$
Output: *Intrusion detection accuracy, Precision, recall,* $F1$ *Score*
1 ***For*** $\forall C$ ***do***
2 *Calculate* $\beta^c = \frac{Q^c}{Q_{benign}}$
3 ***if*** $\beta^c < t$ *then*
4 *apply SMOTE for oversampling the training data (equation 1)*
5 ***endif***
6 ***endfor***
7 *Convert dimensions of training data*
8 *Load* $CNN - BILSTM$ *Model*
9 *Split Training data into Training set and Validationset*
10 *Train model on desired hyperparameter*
11 *Model Validation with validation set*
12 ***For*** *each Attack Class* C *in* D ***Do***
13 *Predict Attack in Testing Data*
14 *Check Model performance on terms of Accuracy, Precision, Recall,* $F1$ *Score*
15 ***endfor***

B. Accuracy

It implies confidence in the classification.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{15}$$

C. Recall / True Positive Rate

It shows retrieval completeness

$$Recall = TP/(TP + FP) \tag{16}$$

D. Precision

Precision is the ratio of true positive samples to samples identified as positive by the system.

$$Precision = TP/(TP + FN) \tag{17}$$

E. F1 Score

It offers an overview of retrieval performance. A greater F1 number suggests that the strategy performs better in terms of Recall and Precision.

$$\text{F1} - \text{ Score} = 2 * (Precision * Recall)/(Precision + Recall) \tag{18}$$

F. Receiver Operating Characteristic Curve (ROC)

A ROC curve is a graph that depicts a classification model's performance and overall classification thresholds. The plot represents the TPR and FPR values as the threshold is varied.

To calculate above indicator the basic attributes of confusion matrix are True Positive (TP) means attack is declared as an attack, True Negative (TN) means non-attack is declared as non-attack, False Positive (FP) means non-attack is declared as attack and False Negative (FN) is attack is declared as non-attack.

4. Experimentation and Results

This section shows the implementation of the proposed model and describes the experimental results. We compare the performance of our model with state-of-the-art methods trained and tested on the mentioned dataset.

4.1. Implementation Setup

The proposed SMOTE integrated CNN-BIDLSTM model was trained using the Nvidia Tesla K80/T4 GPU and 12 GB RAM on the Google Collaborate dataset. A 1D CNN layer and several BIDLSTM layers make up the CNN-BIDLSTM architecture that is combined with SMOTE, and each layer is processed both forward and backward. 1-D Rapid spatial learning is made possible for the provided time series data using CNN. The network's first layer is a 1-D CNN layer, which is followed by a Max Pooling layer to reduce the number of parameters by identifying relevant characteristics and performing computation in the network, and a Batch Normalization layer to standardize operations on input from earlier layers. Two bidirectional hidden layers with the number of neurons listed in Table 1 for each

dataset make up the next layers. Following the first hidden layer, a Reshape layer is used to reshape the output of the prior layer for the following BIDLSTM layer. Between each BIDLSTM layer, a max-pooling and batch normalization layer exists.

We use a dropout layer with a threshold of 0.2 to ensure effective feature learning and reduce overfitting the training set of data. The completely connected dense layer receives the output of this integrated Deep Learning network, which then serves as the final output. The dense layer activation function carries out nonlinear computations using a sigmoid function.

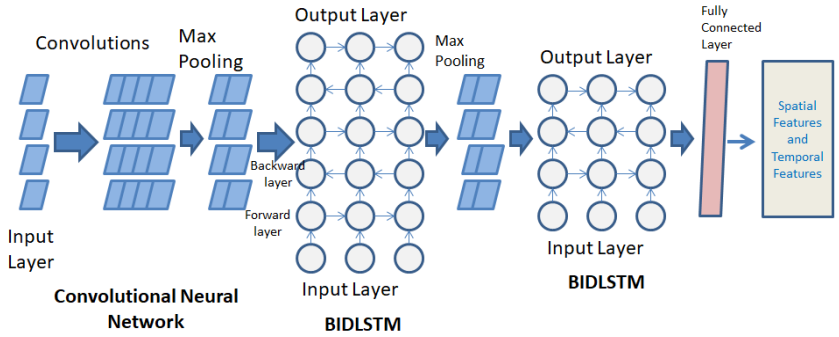

Fig.5. Integrated CNN-BDLSTM deep learning model

For 50 epochs, the ideal batch size was 1024 for the CIC-IDS 2017 dataset and 512 for the NSL KDD and UNSW-NB15 dataset. A Loss function of the sparse-categorical-cross entropy type was employed. The optimizer Adaptive Moment Estimation (Adam) is used to modify the weight of the model.

Table 1. Proposed model hyper parameters

Layer	Type	Output Shape		
		CICDS2017	NSL KDD	UNSWNB15
1	Convolution (ReLU)	128	128	256
2	Max Pooling	128	128	256
3	Batch normalization	128	128	256
4	Bidirectional LSTM	128	128	256
5	Reshape	128	128	256
6	Max Pooling	128	128	256
7	Batch normalization	128	128	256
8	Bidirectional LSTM	64	64	128
9	Dropout	64	64	128
10	Fully connected (sigmoid)	2	2	2

4.2. Dataset

- NSL-KDD: NSL-KDD fixes some issues with the KDD Cup 99 dataset [18]. The dataset includes 41 features, 5 labels, 4 attack types—a Probe attack, a DoS assault, a Remote-to-Login attack, and a User-to-Root attack—as well as benign. More than half of the traffic in the KDDTrain+ set is benign, but U2R and R2L only contribute 0.04 and 0.79 percent, respectively.
- CICIDS 2017: The CIC-IDS 2017 [19] provides researchers with the flexibility to work with contemporary frequent attack variants and is available in pure form. The 5-day traffic record from the CICIDS 2017 dataset includes benign, insider, and external attacks.
- UNSW-NB15: A collection of unprocessed network packets is known as UNSW-NB15 [20, 21]. It contains 2,540,044 entries spread across four CSV files and almost 100GB of data, along with a sizable number of malicious and lawful network objects. Reconnaissance, Worms, Generic, Exploits, Shellcode, Analysis, Fuzzers, Backdoor, and DoS are the ten different labelings (one benign type and nine malicious types), and there are 42 characteristics. Worms, ShellCode, and Backdoor contribute less than 0.1 percent, whereas benign traffic contributes more than half.

4.3. Result and Discussion

To assess the performance of the algorithm, we ran three experiments, which are covered in sections 6.2.1, 6.2.2, and 6.2.3. We first applied the standard LSTM in each experiment, followed by CNN-LSTM, BIDLSTM, and CNN-BiLSTM, and we compared the results with the performance of the SMOTE integrated approach of all the models.

A. CICIDS2017 Dataset

We used all 77 features in the dataset to train the models in this experiment. The dataset was split into training and testing sets using the stratified method. Each sort of attack is used to test the trained binary model. The proposed model's ROC curve is shown in Fig. 5.

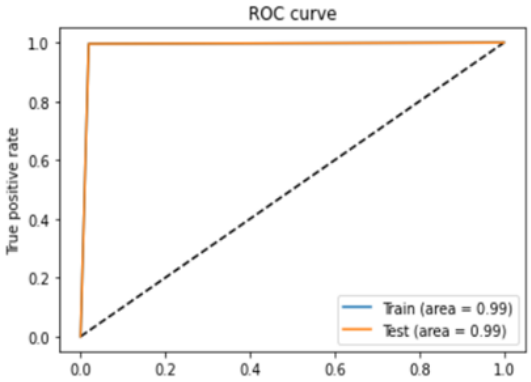

Fig.6. ROC curve for CIC-IDS 2017 dataset

Table 2 lists the accuracy results for each model with and without the SMOTE method (S-Model name). The table shows that, in each instance, the minority classes (Web-BruteForce, Web-XSS, Web-SQL, Bot, Infiltration) respond better to the SMOTE integrated approach. The proposed model's overall training and testing accuracy are 99.97 percent and 99.98 percent, respectively, with an F1-Score of 0.99, a precision of 0.96, and a recall of 0.98. The proposed S-CNN-BiLSTM outperforms each of the other SMOTE integrated models for the attack types of Web-BruteForce, Web-XSS, Web-SQL, Bot, and Infiltration, with the remaining attacks providing similar accuracies to that of the other models.

Table 2. Accuracy of CICIDS2017 dataset

Attack Type	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train	98.15	98.56	98.2	99.02	98.84	98.86	98.7	99.97
Test	98.15	98.17	98.22	98.6	98.37	98.87	98.44	99.98
Benign	98.25	97.93	99.19	98.33	99.11	98.09	99	99.99
PortScan	98.92	99.97	95.13	99.99	94.98	99.84	95	99.95
Web- brute Force	9.8	14.46	17.45	99.4	14	89.44	13.27	**100**
Web- XSS	4.14	4.9	4.44	98.15	6.28	87.88	6.13	**99.84**
Web- Sql	14.28	47.61	47.6	57.14	42.85	52.38	47.6	**95.23**
FTP Patator	99.79	99.84	99.71	99.86	99.92	100	99.8	99.99
SSH- Patator	60.52	95.04	91.02	98.93	97.57	98.79	98.89	96.35
DDoS	98.36	99.01	99.54	99.68	99.91	99.96	99.55	95.35
BOT	37.11	62.47	37.57	62.67	35.78	51.53	37.42	**99.64**
Infiltration	19.44	61.11	22.22	38.8	33.33	47.22	30.55	**97.22**
Heartbleed	90.9	100	100	100	90.9	100	100	100
DoS Goldeneye	96.52	98.59	99.4	99.82	98.08	98.28	99.87	100
DoS Hulk	98.89	99.99	91.24	99.83	99.84	99.8	99.47	99.99
DoS Slowhttptest	99.12	99.23	99.58	99.59	99.27	99.29	99.3	99.96
DoS Slowloris	98	99.49	99.36	99.72	99.24	99.32	99.3	99.91

Table 3. Precision of CICIDS2017 dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train	0.93	0.98	0.97	0.98	0.96	0.98	0.98	0.99
Test	0.93	0.92	0.97	0.94	0.93	0.96	0.93	0.97

Tables 3, Table 4, and Table 5 provide summaries of the dataset's precision, recall, and F1-Scores respectively.

Table 4. Recall of CICIDS2017 dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train	0.98	0.99	0.94	1.00	0.98	1.00	0.99	0.98
Test	0.98	0.99	0.94	1.00	0.98	1.00	0.99	0.98
Benign	0.98	0.98	0.99	0.98	0.99	0.98	0.99	0.98
PortScan	0.99	1.00	0.95	1.00	0.95	1.00	0.95	1.00
Web- brute Force	0.1	0.14	0.17	0.99	0.14	0.89	0.13	**1.00**
Web- XSS	0.04	0.05	0.04	0.98	0.06	0.88	0.06	**1.00**
Web- Sql	0.14	0.48	0.48	0.57	0.43	0.52	0.48	**0.95**
FTP Patator	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SSH- Patator	0.61	0.95	0.91	0.99	0.98	0.99	0.99	1.00
DDoS	0.98	0.99	1.00	1.00	1.00	1.00	1.00	0.96
BOT	0.37	0.62	0.38	0.63	0.36	0.52	0.37	**1.00**
Infiltration	0.19	0.61	0.22	0.39	0.33	0.47	0.31	**0.97**
Heartbleed	0.91	1.00	1.00	1.00	1.00	0.91	1.00	1.00
DoS Goldeneye	0.97	0.99	0.99	1.00	0.98	0.98	1.00	1.00
DoS Hulk	0.99	1.00	0.91	1.00	1.00	1.00	0.99	1.00
DoS Slowhttptest	0.99	0.99	1.00	1.00	0.99	0.99	0.99	1.00
DoS Slowloris	0.98	0.99	0.99	1.00	0.99	0.99	0.99	1.00

Table 5. F1 Score of CICIDS2017 dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train	0.95	0.99	0.95	0.99	0.97	0.99	0.99	0.99
Test	0.95	0.96	0.95	0.97	0.96	0.97	0.96	0.99
Benign	0.99	0.99	1.00	0.99	1.00	0.99	1.00	1.00
PortScan	0.99	1.00	0.98	1.00	0.97	1.00	0.97	0.99
Web- brute Force	0.18	0.25	0.3	1.00	0.25	0.94	0.23	**1.00**
Web- XSS	0.08	0.09	0.09	1.00	0.12	0.94	0.12	**1.00**
Web- Sql	0.25	0.65	0.65	0.99	0.6	0.69	0.65	**0.98**
FTP Patator	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SSH- Patator	0.75	0.97	0.95	0.99	0.99	0.99	0.99	0.98
DDoS	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.98
BOT	0.54	0.77	0.55	0.77	0.53	0.68	0.54	**1.00**
Infiltration	0.33	0.76	0.36	0.56	0.5	0.64	0.47	**0.99**
Heartbleed	0.95	1.00	1.00	1.00	1.00	0.95	1.00	1.00
DoS Goldeneye	0.98	0.99	1.00	1.00	0.99	0.99	1.00	1.00
DoS Hulk	0.99	1.00	0.95	1.00	1.00	1.00	1.00	1.00
DoS Slowhttptest	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
DoS Slowloris	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00

The performance of proposed model on minority attacks (Web-Brute Force, Web-XSS, Web-SQL, Bot, and Infiltration) in CICIDS 2017 can be seen be seen in table 6 with respect to detection accuracy, Recall and F1-Score. The recall is important parameter to distinguish attack class as attack only i.e. it is sensitive to attack predicted as normal, higher the recall better is the model. F1_measure indicates the balance between precision and sensitivity of model. Therefore, F1_Measure is harmonic mean of precision and recall is useful to know the consistency of model. Higher the value better is the model. The performance of proposed model is significantly improved and consistent for all minority attacks on all parameters. The proposed model achieved 95%, 98%, and 57.14% of recall, F1_score and accuracy respectively for Web-SQL attack. The infiltration attack detection accuracy reaches 50.55%, recall is 97% and F1_score is 99%. The proposed model gives 0.98 as the lowest F1_Score on Web-SQL.

As compared with other models we found the SMOTE based CNN-LSTM as next best model in terms of accuracy, recall and F1_score except for Infiltration attack recall is better than S-CNN-LSTM model. The detection accuracy of Web-XSS in Proposed model is more by 1.23% than S-CNN-LSTM model with stable outlook on recall and F1_score.

The proposed model performs better on Web-SQL attack in terms of recall by increase of 38% while keeping stable accuracy and F1_Measure than S-CNN-LSTM model. The proposed model shows significant improvement on BoT with respect to accuracy, recall and F1_measure as 1.03%, 0.37 and 0.23 respectively and on Infiltration attacks with 11.75%, 0.36 and 0.43 as accuracy, recall(S-LSTM) and F1_Score respectively.

Table 6. Performance of S-CNN-BIDLSTM on CICIDS2017 dataset and Performance Improvement

S. N.	Attack Type	Performance			Improvement(Value)		
		Accuracy(%)	Recall	F1_Score	Accuracy(%)	Recall	F1_Score
1	**Web- brute Force**	**99.33**	**1.00**	**1.00**	Stable	Stable	Stable
2	**Web- XSS**	**99.38**	**1.00**	**1.00**	**1.23**	Stable	Stable
3	**Web- Sql**	**57.14**	**0.95**	**0.98**	Stable	**0.38**	Stable
4	**BOT**	**63.70**	**1.00**	**1.00**	**1.03**	**0.37**	**0.23**
5	**Infiltration**	**50.55**	**0.97**	**0.99**	**11.75**	**0.36 (S-LSTM)**	**0.43**
6	**Heartbleed**	**100.0**	**1.00**	**1.00**	Stable	Stable	Stable

B. UNSW-NB15 Dataset

All 49 characteristics from the UNSW-NB15 dataset were used to train the model. The dataset was shrunk to 194 features after encoding. As shown in Table 7, all models performed identically for this dataset, with an overall accuracy of 99.99 percent and an accuracy of 100 percent for every type of attack.

The effect of utilizing SMOTE approach is minimal since in this dataset the class imbalance problem is not seen by a significant margin that would impair the model training. Therefore, the models function well with high precision, recall, and F1 score even without SMOTE.

Table 7. Accuracy of UNSW-NB15 dataset

	w/o Smote Model	Smote integrated Model
Train (train.csv)	100	100
Test (test.csv)	99.9977	99.9988
Benign	99.9957	99.9978
Reconnaissance	100	100
Backdoor	100	100
DoS	100	100
Exploits	100	100
Analysis	100	100
Fuzzers	100	100
Worms	100	100
Shellcode	100	100
Generic	100	100

C. NSL-KDD Dataset

We used the entire dataset's 41 features to train the models in this experiment. 122 features were added to the dataset following feature encoding for a category of variables.

On its KDDTrain+.csv and KDDTest+.csv files, the model is trained and tested. Each sort of attack is used to test the trained binary model. The ROC curve for the suggested model for the NSL-KDD dataset is shown in Fig. 6.

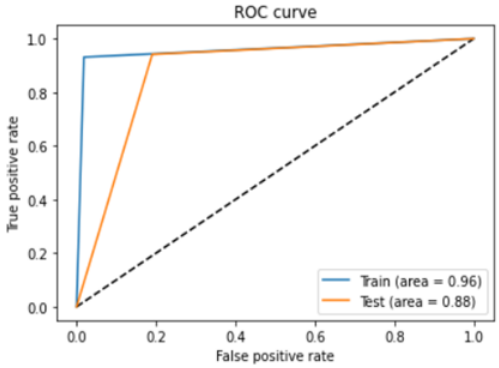

Fig.7. ROC curve for NSLKDD dataset

Table 8 provides an overview of the accuracy results from each of the models with and without the SMOTE technique (S-Model name). Precision, recall, and F1-Scores for the provided model are 0.98, 0.98, 0.98 and 0.95, 0.81, 0.87, respectively, with overall training and testing accuracies of 97.15 percent and 86.64 percent.

Table 8. Accuracy of NSLKDD dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train (KDDTrain$^+$.csv)	95.51	95.5	99.75	98.14	99.73	98.14	99.71	97.15
Test (KDDTest$^+$.csv)	80.87	80.98	78.82	82.18	81.05	81.06	80.3	86.64
Benign	98.53	98.49	99.62	99.2	98.84	98.94	99.24	99.56
DoS	97.71	95.86	97.78	98.4	97.84	97.63	98.68	99.27
Probe	97.6	96.84	96.6	97.16	97.7	97.7	96.34	97.71
R2L	30.95	51.57	34.35	38.32	34.38	39.43	35.02	**55**
U2R	48.73	70.1	56.3	68.06	43.69	63.02	55.46	**70.58**

Table 9. Precision of NSLKDD dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train (KDDTrain$^+$.csv)	0.95	0.95	1.00	1.00	1.00	1.00	1.00	0.98
Test (KDDTest$^+$.csv)	0.97	0.92	0.93	0.96	0.96	0.96	0.92	0.95

Table 10. Recall of NSLKDD dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train (KDDTrain$^+$.csv)	0.95	0.95	1.00	0.98	1.00	0.98	1.00	0.98
Test (KDDTest$^+$.csv)	0.69	0.73	0.68	0.72	0.7	0.7	0.72	0.81
Benign	0.99	0.98	1.00	0.99	0.99	0.99	0.99	1.00
DoS	0.98	0.96	0.98	0.98	0.98	0.98	0.99	0.99
Probe	0.98	0.97	0.97	0.97	0.98	0.98	0.98	0.99
R2L	0.31	0.52	0.34	0.38	0.34	0.39	0.35	**0.55**
U2R	0.49	0.73	0.56	0.68	0.44	0.63	0.55	**0.71**

Table 11. F1 Score of NSLKDD dataset

	LSTM	S-LSTM	CNN-LSTM	S-CNN-LSTM	BiLSTM	S-BiLSTM	CNN-BiLSTM	S-CNN-BiLSTM
Train (KDDTrain$^+$.csv)	0.95	0.95	1.00	0.99	1.00	0.99	1.00	0.98
Test (KDDTest$^+$.csv)	0.8	0.81	0.79	0.82	0.81	0.81	0.81	0.87
Benign	0.99	0.99	1.00	1.00	0.99	0.99	1.00	1.00
DoS	0.99	0.98	0.99	0.99	0.99	0.99	0.99	1.00
Probe	0.99	0.98	0.98	0.99	0.99	0.99	0.98	0.99
R2L	0.47	0.68	0.51	0.55	0.51	0.57	0.52	**0.71**
U2R	0.66	0.84	0.72	0.81	0.61	0.77	0.71	**0.83**

The minority attack classes in NSL-KDD dataset such as U2R and R2L are also giving consistent improvement with respect to accuracy, recall and F1_score. The performance of proposed model compared with other models shows significant improvement. Due to minority samples in few attacks the augmentation applied using SMOTE helps to improve count of samples required to train the model.

On NSL-KDD dataset after the proposed model the SMOTE based LSTM model gives better results. The minority R2L and U2R attacks shows improvement of 3.43 and 0.48 respectively. The recall and F1_score is increased by 0.03 in both on R2L and gives almost stable result on U2R on same parameters.

The proposed model may get over-fit on majority classes which may result in reduced rate by 0.01 on all performance parameters.

Table 12. Performance of S-CNN-BIDLSTM on NSL-KDD dataset

		Performance		
S. N.	**Attack Type**	**Accuracy (%)**	**Recall**	**F1_Score**
1	**Train (KDDTrain$^+$.csv)**	97.15	0.98	0.98
2	**Test (KDDTest$^+$.csv)**	86.64	0.81	0.87
3	**Benign**	99.56	1.00	1.00
4	**DoS**	99.27	0.99	1.00
5	**Probe**	97.71	0.99	0.99
6	**R2L**	55	0.55	0.71
7	**U2R**	70.58	0.71	0.83

5. Conclusions

This study introduces an integrated approach for detecting intrusions based on integrated CNN-BDLSTM model with SMOTE sampling. The main challenge of extraction of useful features from traffic data is addressed by integrating two models. We developed an intrusion detection system that is trained on attacks samples. Along with the UNSW-NB15 and NSL-KDD benchmark datasets, the CIC-IDS2017 dataset, which is enormous and includes the most recent threats and features that are not included in older datasets, is used to evaluate the model. The results show that the overall detection accuracy of the SMOTE combined CNN-BIDLSTM model is quite good reaching 98.85 percent, 99.99 percent, and 86.64 percent for CIC-IDS2017, UNSW-NB15, and the NSL-KDD, respectively. The proposed model gives ROC area coverage of 99% on both training and testing of CICIDS 2017 dataset.

The extensive experimentation and significance tests indicate that the suggested approach performs better than the deep learning models LSTM, CNN-LSTM, and BIDLSTM. Additionally, the model performs better for the minority attack classes Web-BruteForce, Web-XSS, Web-SQL, Bot, Infiltration and R2L, U2R in the CIC-IDS2017 and NSL-KDD datasets, respectively. The performance of model is significantly improved and or stable in minor attacks.

Further study will concentrate on how to improve the accuracy of infiltration, web-sql, and bot attack types in addition to optimizing parameters and architecture to increase accuracy. Additionally, research can be done on newer, more intelligent data augmentation techniques. Additionally, fast training models can be investigated to address the issue of increased computational requirements and training time brought on by the extended training set.

References

[1] Aleesa, Ahmed, M. O. H. A. M. M. E. D. Younis, AHMED A. Mohammed, and N. Sahar. "Deep-intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques." Journal of Engineering Science and Technology 16, no. 1 (2021): 711-727.

[2] Shende, Supriya, and Samrat Thorat. "Long Short-Term Memory (LSTM) Deep Learning Method for Intrusion Detection in Network Security." *International Journal of Engineering Research and* 9 (2020).

[3] Yin, Chuanlong, Yuefei Zhu, Jinlong Fei, and Xinzheng He. "A deep learning approach for intrusion detection using recurrent neural networks." *Ieee Access* 5 (2017): 21954-21961.

[4] You, Wei, Changqing Shen, Dong Wang, Liang Chen, Xingxing Jiang, and Zhongkui Zhu. "An intelligent deep feature learning method with improved activation functions for machine fault diagnosis." *IEEE Access* 8 (2019): 1975-1985.

[5] Ahsan, Mostofa, and Kendall E. Nygard. "Convolutional Neural Networks with LSTM for Intrusion Detection." In *CATA*, vol. 69, pp. 69-79. 2020.

[6] Sun, Pengfei, Pengju Liu, Qi Li, Chenxi Liu, Xiangling Lu, Ruochen Hao, and Jinpeng Chen. "DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system." *Security and communication networks* 2020 (2020).

[7] Su, Tongtong, Huazhi Sun, Jinqi Zhu, Sheng Wang, and Yabo Li. "BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset." *IEEE Access* 8 (2020): 29575-29585.

[8] Imrana, Yakubu, Yanping Xiang, Liaqat Ali, and Zaharawu Abdul-Rauf. "A bidirectional LSTM deep learning approach for intrusion detection." *Expert Systems with Applications* 185 (2021): 115524.

[9] Yang, S. U. "Research on network behavior anomaly analysis based on bidirectional LSTM." In *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pp. 798-802. IEEE, 2019.

[10] Li, Aichuan, and Shujuan Yi. "Intelligent Intrusion Detection Method of Industrial Internet of Things Based on CNN-BiLSTM." *Security and Communication Networks* 2022 (2022).

[11] Praanna, K., S. Sruthi, K. Kalyani, and A. Sai Tejaswi. "A CNN-LSTM model for intrusion detection system from high dimensional data." *J. Inf. Comput. Sci* 10 (2020): 1362-1370.

[12] Hindy, Hanan, David Brosset, Ethan Bayne, Amar Kumar Seeam, Christos Tachtatzis, Robert C. Atkinson and Xavier J. A. Bellekens. “A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets.” *ArXiv* abs/1806.03517 (2018).

[13] Liu, Hongyu, and Bo Lang. "Machine learning and deep learning methods for intrusion detection systems: A survey." *applied sciences* 9, no. 20 (2019): 4396.

[14] Ferrag, Mohamed Amine, Leandros Maglaras, Helge Janicke, and Richard Smith. "Deep learning techniques for cyber security intrusion detection: A detailed analysis." *6th International Symposium for ICS & SCADA Cyber Security Research* (2019), pp. 126-136.

[15] Hindy, Hanan, Robert Atkinson, Christos Tachtatzis, Jean-Noël Colin, Ethan Bayne, and Xavier Bellekens. "Utilising deep learning techniques for effective zero-day attack detection." *Electronics* 9, no. 10 (2020): 1684.
[16] Ieracitano, Cosimo, Ahsan Adeel, Francesco Carlo Morabito, and Amir Hussain. "A novel statistical analysis and autoencoder driven intelligent intrusion detection approach." *Neurocomputing* 387 (2020): 51-62.
[17] Kim, Jin-Young, Seok-Jun Bu, and Sung-Bae Cho. "Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders." *Information Sciences* 460 (2018): 83-102.
[18] Tavallaee, Mahbod, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. "A detailed analysis of the KDD CUP 99 data set." *IEEE symposium on computational intelligence for security and defense applications*, pp. 1-6. IEEE, 2009.
[19] Sharafaldin, Iman, Arash Habibi Lashkari and Ali A. Ghorbani. "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization." *International Conference on Information System Security and Privacy*, ICISSP, (2018): 108-116.
[20] Kumar, Vikash, Ayan Kumar Das, and Ditipriya Sinha. "Statistical Analysis of the UNSW-NB15 Dataset for Intrusion Detection." *Advances in Intelligent Systems and Computing*, (2019), 279–94.
[21] Kumar, Vikash, Ditipriya Sinha, Ayan Kumar Das, Subhash Chandra Pandey, and Radha Tamal Goswami. "An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset." *Cluster Computing* 23, no. 2 (2020): 1397-1418.

Authors' Profiles

Shrinivas A. Khedkar is from Mumbai, India and born in 1988. He has completed his Bachelor of Technology in IT in 2011 from Dr. BATU Lonere,MH, India and Master of Technology in Computer Engineering in 2014 from SGGS IEand T, Nanded,MH, India. His research interest includes cyber security, machine learning and deep learning.

He joined the VJTI Mumbai, in 2015. He is working as an Assistant Professor in Computer Engineering to undergraduates. He is life member of ISTE.

Dr. Madhav Chandane was born in Nanded, India, in 1972. He received the B.E. degree in Computer Science and Engineering from SGGSIE and T, Nanded, India, in 1997, the M. Tech and Ph.D. degree in Computer Engineering from VJTI, Mumbai, India in 2004 and 2013 respectively.

He worked as Lecture in VJTI Mumbai from 2000 to 2007. From 2008 to 2010, he was Assistant Professor at same institute. Since 2011, he has been Associate Professor with Computer Engineering and IT department at VJTI Mumbai. He has published research papers in reputed journals as well as in National and International conference.

Rasika Gawande was born in 1999 in Mumbai, India. She received the B. Tech degree in Computer Engineering from VJTI, Mumbai, India in 2022.

How to cite this paper: Shrinivas A. Khedkar, Madhav Chandane, Rasika Gawande, "Integrated Spatial and Temporal Features Based Network Intrusion Detection System Using SMOTE Sampling", International Journal of Computer Network and Information Security(IJCNIS), Vol.16, No.2, pp.14-27, 2024. DOI:10.5815/ijcnis.2024.02.02