IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 34, No. 3, June, 2020, pp. 323-330
Journal homepage: http://iieta.org/journals/ria

Predictive Model for Network Intrusion Detection System Using Deep Learning

Venkatramaphanikumar Sstla[1*], Venkata K.K. Kolli[1], Lakshmi K. Voggu[1], Ramyasri Bhavanam[1], Sasikala Vallabhasoyula[2]

[1] Department of CSE, VFSTR Deemed to be University, Vadlamudi, Guntur 522213, Andhra Pradesh, India
[2] Department of IT, VNITSW, Guntur 522006, Andhra Pradesh, India

Corresponding Author Email: drsvpk_cse@vignan.ac.in

https://doi.org/10.18280/ria.340310

Received: 15 February 2020
Accepted: 5 May 2020

Keywords:
IDS, NIDS, support vector machine, deep neural networks, NSL-KDD

ABSTRACT

Given the recent COVID-19 situation, many organizations and companies have asked their employees to work from home by connecting to their on-premises servers. This situation may continue a much more extended period in the future, thereby opening more threats to confidentiality and security to the information available in the organizations. It becomes of hell of a task for network administrators to counter the threats. Intrusion Detection Systems are deployed in firewalls to identify attacks or threats. In preset modern technologies, Network Intrusion Detection System plays a significant role in defense of the network threat. Statistical or pattern-based algorithms are used in NIDS to detect the benign activities that are taking place in the network. In this work, deep learning algorithms have developed in NIDS predictive models to detect anomalies and threats automatically. Performance of the proposed model assessed on the NSL-KDD dataset in the view of metrics such as accuracy, recall, precision, and F1-score. The experimental results show that the proposed deep learning model outperforms when compared with existing shallow models.

1. INTRODUCTION

Intrusion detection is one of the vital concepts in computers and the servers' security. The process of examining the events that occurred inside the network, and figuring out the intrusions and threats for security is a challenge for any organization [1, 2]. An Intrusion Detection System (IDS) [3] analyses and monitors network traffic to protect a system from malicious activities or network threats. But intrusions and misuse have always threatened the secured data communication over networks. Recent world notorious hackers are using different types of attacks for hacking the valuable information. IDSs are classified into majorly of three types, such as Network Intrusion Detection System, Host-based Intrusion Detection System, and Hybrid-based Intrusion Detection System. All these intrusion detection systems and algorithms help in the detection of various kinds of attacks.

NIDS can access the information while traveling through the network. NIDS helps in detecting security threats at the network level. NIDS is installed at a place in the network (example: router) from where it is possible to watch the traffic in the network. In HIDS, the encrypted information is decrypted and then only it can be used for processing, A HIDS is placed on a server or specific computer (example: host). HIDS monitors the status of that key system and identifies when an intruder creates, deletes, or modifies the files. In host systems, HIDS detects local events. Hybrid based IDS is the combination of both NIDS and HIDS. NIDS has more monitoring capacity when compared to HIDS. IDS are also classified based on the detection approach, signature-based detection, anomaly-based detection, and reputation-based detection.

Signature-based detection is used for recognizing bad patterns (example: malware). Anomaly based detection is used for detecting deviations from good traffic. Reputation-based detection is used to find potential threats in accordance with the reputation. NIDS gives the best solution to network security when compared to other network defence technologies. The IDS extension is Intrusion Prevention System (IPS). IDS is used for identifying or detecting the threats (monitors), as well as IPS is used for preventing the threats (controls) and alters the traffic flow. IPS are of two types: Active and Inline. IDS is efficient in monitoring the network and prevents alterations to directories and files. The only disadvantage of IDS is the inability to detect the source of the attack. In case of any such attack, it locks the whole network. Some characteristics of IDS are the ability to identify the significant derivations from typical system behaviour. IDS must survive with the change in environment and behaviour of the system, as new technologies and applications are continuously included. NIDS detects the threats which had failed to perform any kind of serious damages, and it will respond quickly to the attacks and threats in real-time. NIDS is faster than HIDS [4].

In the recent past, Machine learning (ML) algorithms perform well in detecting the treats in IDS. ML algorithms are classified into three types: Supervised, Unsupervised, and Reinforcement learning. Supervised learning uses the label data for creating a model, and then they are categorized into two more methods, such as Classification and Regression. Suppose if there are two classes like Yes - No, True - False (categorical variables) classification is used in such cases. Random Forests, Decision Trees, Logistic Regression, and SVM are some of the algorithms used in classification.

Regression is used to map the relationship between continuous variables. Unsupervised learning does not use any label or classified data, and this learning arranges the input data into group of features or objects with similar patterns. They are two categories in this; one is Clustering and the other is Association. In this Reinforcement learning, the learning agent will work according to the feedback, getting the penalties for wrong actions and rewards for right actions. The main objective of the learning agent is to get more reward points, and performance can be improved based on the reward points of an agent. Recent developments in hardware lead to the evolvement of Deep Learning (DL) methods such as Convolutional Neural Networks (CNN), Recurrent Neural Network (RNN), LSTM, etc. In Deep Neural Networks (DNN) performs well on several hidden layers. DNN helps in dealing with applications Detecting pedestrians, helps in decreasing accidents), Aerospace and Defence, Automated Driving, Medical Research, Industrial Automation, Electronics.

Known attacks can be easily identified, but unknown classes of attacks are very harder to detect. So in this work, DNN classifiers are proposed, which not only learns but also adapts itself to the patterns which are not defined earlier. DNN model is used to develop effective and flexible IDS to identify, detect, and classify unpredictable cyber-attacks. The continuous change within the evolution of attacks and network behaviour makes it necessary to compute and evaluate different datasets. The DNN performs well on NSL-KDD dataset with different epochs (50-500) with different learning rates [0.01-0.5]. NSL-KDD dataset is used to evaluate the proposed model in this work. A detailed literature survey is presented in section 2. The proposed methodology is presented in section 3. Experimental results are discussed in section 4. Finally, the paper concludes with the conclusion and future scope.

2. RELATED WORK

Intrusion Detection System is a defence system that is used to detect intrusions or intruders who are trying to gain unauthorized access to the network. Generic architecture of IDS is given in Figure 1. IDS might be a software application or a hardware device that uses the intrusion signatures, to detect and analyse the in-bound and out-bound traffic in the network for unusual activities.

Firewalls can be used in IDS and that are used to prevent any malignant attack on the network or on any system. Firewalls basically work as filters which obstruct or stop any kind of data that cause a threat to the network or the system. Firewalls can either monitor the content of the whole packet or monitor only some content of the packet. IDS will monitor request data and then check if it contains normal or attack signatures, if it contains attack signatures then request will be dropped. IDS will be trained with all possible attacks' signatures with machine learning algorithms and then generate train model, whenever new request signatures arrived then this model applied on new request to determine whether it contains normal or attack signatures. To avoid all attacks IDS systems has developed which process each incoming request to detect such attacks and if request is coming from genuine users then only it will forward to server for processing, if request contains attack signatures then IDS will drop that request and log such request data into dataset for future detection purpose. To detect such attacks IDS will be prior train with all possible attacks' signatures coming from malicious user's request and then generate a training model. Upon receiving new request IDS will apply that request on that trained model to predict the class whether the request belongs to normal class or attack class. To train such type of models, several data mining classifications and prediction algorithms will be used. Intrusion Detection systems are of three types (based on working method) HOST-based (HIDS), NETWORK-based (NIDS) [5] and Hybrid-based detection systems.

HIDS works on each individual device on the network. A HIDS monitors both in-coming and out-going packets from the device and will alert the administrator. A HIDS is placed on a server or specific computer (e.g.: host). HIDS can monitor the status of that specific key system files and recognizes when an intruder creates, deletes, or modifies the files. HIDS [6] can detect local events on computers or host systems and architecture of HIDS is given in Figure 2.

HIDS is a method, which is used for providing security for networks as well as computers. The main operation of a HIDS will generally depend on the reality that hackers or intruders who want to inherent the computers which they have hacked. Intruders were installing software to establish their 'ownership' on the computer. This will help the intruders to access each activity on the computer, which is going to take place in the future. HIDS will detect such types of modifications and report its findings to the administrator. In HIDS, sensors commonly consist of a software agent. An example of HIDS is OSSEC (Open Source HIDS Security).

NIDS is placed at a strategic point within the network to monitor the traffic. NIDS [7] helps to detect the security threats which are related to the network. NIDS considers the extension of the network and new attacks, which may come from several resources. A NIDS is present in a device that is connected to the segment of the network in any organization. NIDS is installed at a place in the network (example: host). NIDS is an individualistic platform, which is used to identify the intrusions by examining the traffic over the network, and it also monitors the multiple hosts. NIDS will obtain access to network traffic by connecting a bridge to the network switch and router. In NIDS, sensors are located at the choke points in the network borders or at the demilitarized zone (DMZ). These sensors will capture all the network traffics and identifies the malicious activities in individual packets. Example of NIDS is snort and main advantage of NIDS is that, it can be easily deployed at reasonable cost. It is not necessary to load the NIDS on each system. Architecture of NIDS is presented in Figure 3.

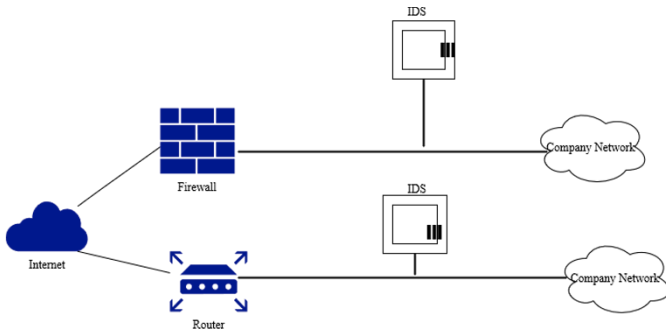

Figure 1. Intrusion detection systems

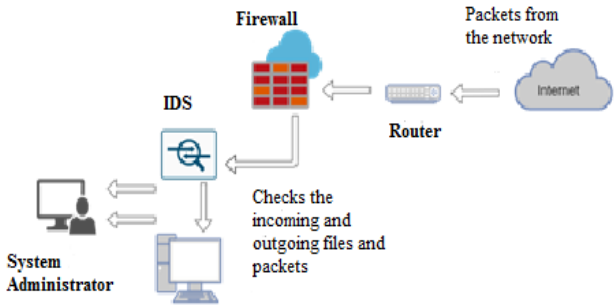

Figure 2. Host-based intrusion detection systems

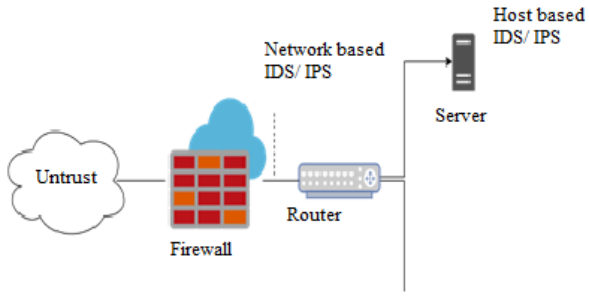

Figure 3. Network intrusion detection systems

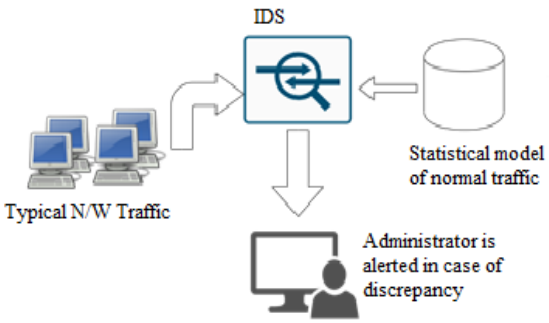

Figure 4. Anomaly intrusion detection systems

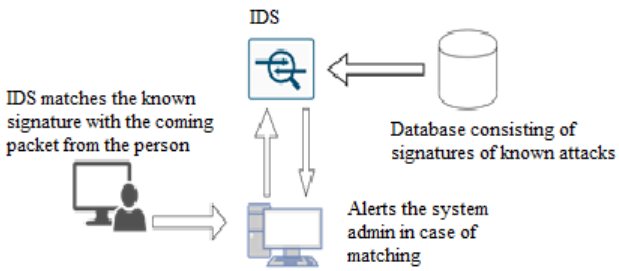

Figure 5. Signature intrusion detection systems

Hybrid-based Intrusion Detection System: Both Network and Host-based IDS [3] are used simultaneously. Detection methods are of two types anomaly-based [8], Signature-based (misuse). Anomaly based detection is the process of examining the patterns in a dataset whose behaviour is abnormal used to map the unknown patterns. These abnormal behaviors are termed as outliers or anomalies. This detection provides critical and significant information in many applications. It consists of data about the statistical model of network traffic, which contains the small print about protocols used for the traffic, bandwidth, ports, and devices used. It continuously looking at the traffic and matches it with the statistical model. In case of any anomaly, alert will be sent to administrator. The main advantage of this Anomaly detection is to detect unique attacks.

Anomaly based IDS, uses Machine Learning ML to create a trustful model [9, 10]. Architecture of Anomaly IDS is given in Figure 4. Signature based IDS is used to detect the attacks based upon some specific patterns and previously known malignant instruction sequence which is used by malware. Generic architecture of Signature based Intrusion Detection System is presented in Figure 5. Specific patterns like binary numbers or number of bytes in network traffic. The patterns which are detected in IDS are termed as signatures. It can easily identify the attacks of known patterns or known signatures, the data is compared with the known attacks signature, if in case of match found an alert will be sent. But for some new attacks like unknown signatures are difficult to detect. It produces more accurate results. Anomaly based IDS [11] has the best generalized property as compared to signature based IDS. These models are going to be trained consistent with the hardware configurations and applications. NIDS provides the best solution to network security when compared to other traditional technologies. Combination of NIDS and other technologies like Artificial Neural Network (ANN) results in increase of detection and prediction rates.

xSDN based NIDS [12] developed using Restricted Boltzmann Machine (RBM) for feature selection and SVM used as the classifier. NSL-KDD dataset is used to evaluate the performance of the model., A Deep Learning approach [13, 14], feed-forward fully connected DNN (Deep Neural Network) is used to train a NIDS (Network Intrusion Detection System). They also used Autoencoder to classify and detect attack traffic in the absence of malicious traffic and ISCX IDS 2012 dataset, and CIC IDS 2017 datasets are for performance evaluation. Convolutional Neural Network (CNN) [15] is used on KDDCup'99 and NSL-KDD datasets to gauge the results, analyse the effectiveness of CNN for ID(Intrusion Detection), model the network traffic events as statistic of TCP/IP packets. THE multiclass SVM [6] model is used in the integration of the Decision Tree-based SVM model to implement IDS and evaluated performance on the KDD Cup'99 dataset [16, 17].

NARX algorithms [18] used with the feed-forward neural network give an accuracy of 97.9% when tested on the KDD Cup'99 dataset [19]. Three data mining based frameworks for NIDs [20] are proposed. They applied the Random Forests algorithm in tracing misuse, anomaly, and hybrid detection to deal with the issues of rule-based systems. By learning over training data, the Random Forests Algorithm RFA can build the patterns automatically rather than coding rules manually. Selvamani and Selvi [21], Anbalagan et al. [22] used simple ANN and Hybrid ANN for anomaly detection. Simulated Annealing Neural Network (SA), Self-Organizing Maps (SOM), Back Propagation Neural Network (BPNN), SVM algorithms are used in a simple approach for anomaly detection. CNN is applied on the NSL-KDD dataset to get detection rate [10] and translate a 1-D array from network flow to a 2-D array to feed into Convolution Neural Network(CNN).

A novel Deep Learning model is proposed in the studies [23, 24] to enable Network Intrusion Detection System (NIDS) within modern networks. They used the combination of Shallow Learning (SL) [10] and deep learning [25, 26], capable of analysing a wide range of traffic in the network [27-29]. They proposed NDAE (Non Symmetric Deep Auto-encoder) and evaluated the model with the help of GPU-enabled Tensor Flow and used NSL-KDD [30] and KDD Cup'99 datasets [8] to evaluate the performance. Elimination

of useless input leads to possibly faster, simplification of the problem, and more accurate detection in IDS [15, 25]. There are two types of feature ranking, one method is independent of the modelling tool, and another method is specific to SVMs. The performance of IDS [8] is compared with different NN (neural network) [18] classifiers. The performance of four proposed models is evaluated on full-featured KDD Cup'99 and reduced featured KDD Cup'99 dataset. By comparing the four classifiers, Feed Forward Neural Network (FFNN) technique gives more efficiency than PNN, RBNN and GRNN for R2L attack, DoS attack and U2R attack. Used DGSOT (Dynamically Growing Self-Organizing Tree) [31, 32] algorithm for clustering. Compared random selection and Rocchio Bundling technique in terms of training time gain and accuracy using a benchmark dataset.

Genetic Algorithm [33, 34] is used to detect different types of network intrusions. Used the KDD Cup'99 dataset [35] to evaluate the performance. A novel methodology [36-38] is proposed, which combines the benefits of MAPE-K frameworks and self-taught learning to deliver a self-adaptive, scalable, autonomous misuse IDS. The MAPE-K control loop is used as a reference model in Automatic computing. This methodology is used to increase the ADR of the Intrusion Detection System [39] to 73.37%. Used NSL-KDD [30] and KDD Cup'99 [40] datasets to evaluate the performance. Genetic Algorithm (GA) [39] based approach is used to detect network intrusions. Used the KDD benchmark dataset and linear structure rule to evaluate the performance. Reinforcement learning is employed in deep auto-encoder DAEQ-N attempts [41] to get the maximum prediction accuracy rate in online learning systems. The Q-learning agent of the proposed NID can predict the anomalies by learning methodology. Different types of Intrusion Detection System (IDS) and IDS tools [7] are discussed.

3. PROPOSED METHOD

Deep Convolutional Neural Networks (DCNN) is intended to try to behave like the human visual system. Consequently, DCNNs have made great achievements in the various fields of computer vision and they are stacked with alternate convolutional and pooling layers. The convolutional layers are used to extract features, and the pooling layers are meant to enhance the generalization among the features [42]. CNNs works well on two-dimensional data, and then well designed to detect the intrusions and generic architecture of CNN is given in Figure 6. Simple Neural Networks have only one input and output layer with some hidden layers, if there are many hidden layers it is termed as Deep Neural Networks. DCNN helps in dealing with large number of datasets and it is influenced by characteristics of biological NN to incorporate the intelligence in the proposed method.

Each node relates to other neurons through a link or bridge. Each link has weight. Input signals information can be represented by this weight. It acts as useful data for nodes to unravel the specific problem. Each node will have an indoor state known as an activation function. The output which came after combining both activation function and input signals can be given as input to other nodes. CNN is trained under supervised learning. After getting the output, this output vector will be compared with the desired output vector, which is termed as target. If suppose an error is generated, we need to update the weights until the actual output is matched with the target value. The input layer of NN (Neural Networks) consists of input neurons, helps to send the input data, and it doesn't take any input from previous layers. So, the input layer being constituted of passive neurons. Hidden Layer is in the middle of the input layer and the output layer. All the calculations, finding probability, patterns, and hidden features can be done by the hidden layer, by using several activation functions. Several activation functions can be used at hidden layers, Such as Softmax, Sigmoid, ReLu, and Tan hyperbolic. These activation functions are given as follows:

$$Sigmoid = \frac{1}{1 + e^{-x}} \tag{1}$$

$$Softmax = \frac{e^{x}}{sum(e^{x})} \tag{2}$$

$$Relu\ y = \max(0, \infty) \tag{3}$$

$$Tanh \frac{2}{1 + e^{-2x}} - 1 \tag{4}$$

Activation functions have a major effect on the NN's ability to converge and convergence speed, or it might prevent NNs from converging in the first place. Output layer- The weighted total which is obtained from $\sum_{i=1}^{n} w_i * x_i + \mathrm{b}$ is passed as an input to an activation function to produce the output. The final output (obtained output) is equalized to the target value. If suppose it equals, then no need to perform backpropagation. If not equals, then we need to find the error by using loss function, this loss function is used to measure the error. There are several loss functions such as Regression, Binary classification, and Multiclass classification loss functions. In the backward pass, weights will be updated. After updating the weights, then again start performing from the first step, this needs to be done until output equals to target value. In this work, we are using 'Binary-Cross Entropy' as a loss function.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\big(p(y_i)\big) + (1 - y_i) \cdot \log(1 - p(y_i)) \tag{5}$$

Here, y is the label and P(y) is the predicted probability. CNN has the capability of analysing a large amount of information or data. Feed Forward Neural Networks allows signals to travel in only one direction that is from input to output and tends to be straight forward networks that associate inputs with outputs. FFNN is classified into two types Single-Layer and Multi-Layer FFNN. Single-Layer FFNN is used to have one input layer and one output layer in which Multi-Layer FFNN consists of input layer, output layer and hidden layer. Architecture of DCNN is given in Figure 7. There are not any feedback loops within the network, hence it's unidirectional and can be used for simple neural networks. Known attacks can be easily identified but the unknown classes of attacks are very harder to detect, so we are using deep learning or DCNN, DCNN not only learns but adjust itself to the patterns which are not defined earlier. DCNN model is used to develop effective and flexible IDS to identify, detect and classify unpredictable cyber-attack. The continuous change in the evolution of attacks and network behaviour makes it necessary to compute and evaluate different datasets. The DCNN model performed well on NSL-KDD dataset with different epochs (50-500) with different learning rates [0.01-0.5].

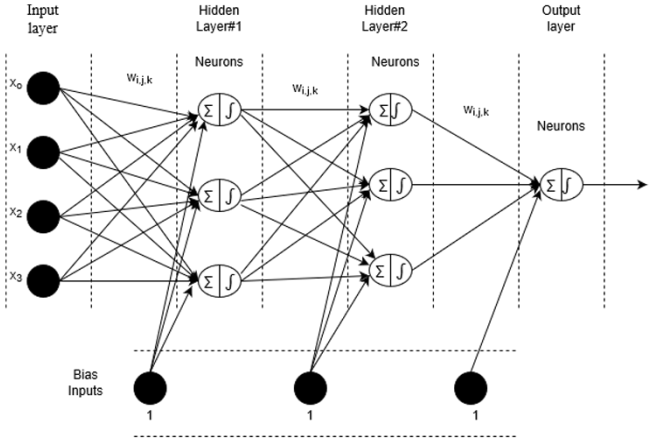

Figure 6. Generic architecture of a CNN

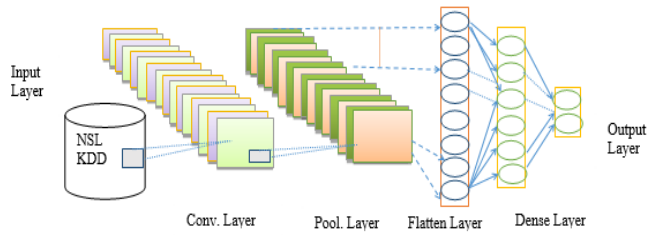

Figure 7. Architecture of the proposed DCNN

NSL-KDD [28] dataset is suggested to solve inherent problems of KDD`99 dataset. Deep learning algorithms have high running time include both training and testing time than regular shallow models. Deep learning models have learnable and hyper parameters. Hyper-parameters are set before training and then learnable parameters are computed during the model training. Deep learning models learnt from raw data, but regular shallow models learnt during future engineering. Deep learning is a division of machine learning, and with the characteristics of deep learning models they are obviously better than traditional machine learning methods. The differences between shallow models and deep models are mainly reflected in the following aspects. Deep learning models have stronger fitting abilities than regular machine learning models.

4. EXPERIMENTAL RESULTS & DISCUSSIONS

Performance evaluation of the proposed method is carried on the NSL-KDD dataset. IDS [8] can be built only when there is an availability of an efficient and effective data set. The NSL-KDD [28] data set are intended to resolve the limitations of the KDD'99 dataset. The performance of the proposed DCNN and SVM are presented in this section. The main advantages of this data set are training set doesn't consist of any redundant records, so that the classifier no longer produces the biased result. The number of records from each selected difficult group is inversely proportional to the percentage of records in the original KDD data set. The 43 attributes of the NSL-KDD dataset are given in the following Table 1. In this work, Chi-Square -based [21] feature selection method is used to reduce the dataset size. These feature selection approaches are used to remove the irrelevant data by holding the necessary and discriminating features from the dataset. In the above dataset records, we can see some values are in string format such as tcp, ftp_data and these values are not important for prediction and these values will be removed during pre-processing phase. All attack names will not be identified by algorithm if it is given in string format, so we need to assign numeric value for each attack. To evaluate the performance of the proposed method, a sample dataset of with 12440 records are used. Out of these 12440, 9950 are used training and remaining 2490 are used for testing. Performance of the proposed method is assessed in the views of precision, recall, F-score and accuracy.

SVM is a popular algorithm mainly used for classification in ML. SVM consists of different kernels, to predict the class labels. SVM kernel functions such as RBF, polynomial, sigmoid and linear are applied on the dataset. Linear kernel is used for simple problems; non-linear kernels such as Radial Basis Function (RBF) and polynomial are used in the classification of complex problems. RBF kernel is used when there is no knowledge on which type of data we are using. RBF is used in different kernelized learning algorithms. RBF is homogeneous, i.e., value of RBF depends upon the distance between the elements or points. Out of 2490 cases 1210 cases are predicted correctly, but 370 cases are predicted under false negative category i.e., predicted as false when they are true and other 910 cases are predicted as true when they are false. Performance evaluation with SVM linear kernel is given in Table 2.

Out of 2490 test samples, 1200 samples are predicted as true positive, and only one sample was under false negative within the remaining 1280 samples, 910 samples are recognized under true negative, and 370 samples are recognized under false positive. In total, the model has recognized 2210 samples correctly out of 2490 samples with an accuracy of 85% and details presented in Table 3.

Using RBF kernel with gamma value "1", out of 2490 test samples, only 1320 samples are correctly classified with an accuracy of 54.21%. With SVM, Polynomial as Kernel, gamma value 'auto'&1 and with degree=3; out of 2490 test samples, 1280 samples are correctly classified and achieved an accuracy of 51.4%. With the same Polynomial Kernel, when gamma as None and degree as six, classification accuracy has fallen to 49%. In the earlier case, the method has failed to recognize samples under true positive but in the other case, and utmost true negative samples were failed to be recognized. When we are evaluating the proposed model using Deep Neural Network as a classifier number of features has been reduced to 13 using various feature selection approaches. To realize the computational challenges, we have divided the data into batches.

Total features are 38, after reducing the feature set through the feature selection process is 13. In DCNN, there is a necessity to pass the entire data set several times through the networks, because updating the weights in single epoch results in underfitting of the model. As the quantity of epochs increases, as well as a greater number of times, the weights are updated, and therefore the curve goes from underfitting to optimal and then may also get into overfitting also. An increase in the batch size results in the more requirement of memory space. The batch size must be more or adequate to one and lesser than or adequate to the number of samples within the training data set. Various activation functions, such as RELU, Sigmoid, Softmax, Tanh. The experimentation continued to identify the correct activation function to be used at input, hidden, and output layers. RELU takes less time to train the model. The number of epochs used is 4, batch_size is

32. Firstly, Relu is used as an activation function at the input and hidden layers. Experimentation carried with other activation functions at the output layer has yielded the accuracy following Table 4.

Table 1. Attributes of NSL-KDD dataset

S.No	Attribute Name	Description
1	Duration	Length of the connections (no.of seconds)
2	Protocol-type	Type of protocol example: Tcp, Udp.
3	Service	Services like HTTP, telnet, etc. Network service at the destination.
4	Flag	Either the status of connection is normal or error.
5	Src_bytes	The number of data bytes from source to destination.
6	Dst_bytes	The quantity of data bytes from destination to source.
7	Land	If the connection is to/from the same host then land is 1, Otherwise: 0
8	Wrong_fragment	Number or quantity of 'wrong' packets.
9	Urgent	Number or quantity of 'urgent' packets
10	Hot	Number or quantity of 'hot' indicators.
11	Num_failed_logins	Number of login attempts that are failed.
12	Logged_in	If successfully logged in-1, otherwise-0.
13	Num_compromised	Quantity of 'compromised' conditions.
14	Root_shell	If root shell is successfully reached then it is-1, Otherwise- 0.
15	Su_attempted	If, command 'su root' is attempted successfully then it is-1, otherwise- 0.
16	Num_root	Quantity of 'root' accesses.
17	Num_file_creations	Quantity of 'file creations'.
18	Num_shells	Quantity of shell prompts.
19	Num_access_files	Number of operations performed on accessing and controlling the files.
20	Num_outbound_cmds	Number of out-bound commands in a FTP session.
21	Is_host_login	If login belongs to 'host' list then it is-1, otherwise-0.
22	Is_guest_login	If the login belongs to 'guest' list then it is-1, otherwise-0.
23	Count	Count indicates the number of connections to the same host, as the present connection in the last 2 seconds.
24	Srv_count	Quantity of connections to the similar service, as the present connection in the last 2 seconds.
25	Serror_rate	Percentage (%) of connections which have "SYN" errors.
26	Srv_serror_rate	Percentage of connections which have "SYN" errors
27	Rerror_rate	Percentage (%) of connections which have "REJ" errors.
28	Srv_rerror_rate	Percentage of connections which have "REJ" errors.
29	Same_srv_rate	It defines the percentage of connections that have been made to the same service.
30	Diff_srv_rate	It defines the percentage of connections that have been made to the different services.
31	Srv_diff_host_rate	It defines the percentage of connections with different hosts.
32	Dst_host_count	Quantity of connections with the same host as the present connection in the last 2 seconds.
33	Dst_host_srv_count	Quantity of connections to the same service as the present connection in the last 2 seconds.
34	Dst_host_same_srv_rate	Percentage of connections that have been made to the same service.
35	Dst_host_diff_srv_rate	Percentage of connections that have been made to different services.
36	Dst_host_same_src_port_rate	Percentage of connections with port value of the same source to the destination.
37	Dst_host_srv_diff_host_rate	Percentage of connections to the different hosts.
38	Dst_host_serror_rate	Percentage (%) of connections which have "SYN" errors.
39	Dst_host_srv_serror_rate	Percentage (%) of connections which have "SYN" errors.
40	Dst_host_rerror_rate	Percentage (%) of connections which have "REJ" errors.
41	Dst_host_srv_rerror_rate	Percentage (%) of connections which have "SYN" errors.
42	Dst_host_diff_src_port_rate	Percentage of connections with port value of the different source to the destination.
43	Label	Yes(1) or No(0)

Table 2. Classification report with SVM linear Kernel

	Precision	Recall	F1-Score	Support
0.0	0.49	1.00	0.65	1210
1.0	0.00	0.00	0.00	1280
Accuracy			0.49	2490
Macro Average	0.24	0.50	0.33	2490
Weighted Average	0.24	0.49	0.32	2490

Table 3. Classification report with SVM RBF Kernel

	Precision	Recall	F1-Score	Support
0.0	0.76	0.99	0.86	1210
1.0	0.99	0.71	0.83	1280
Accuracy			0.85	2490
Macro average	0.85	0.85	0.85	2490
Weighted average	0.88	0.85	0.84	2490

Table 4. Classification with DCNN-Relu as activation

Epochs	Relu	Sigmoid	Softmax	Tanh
10	37%	83%	48%	45%
100	46%	96%	48%	39%

Table 5. Classification with DCNN-sigmoid as activation

Epochs	Relu	Sigmoid	Softmax	Tanh
10	55%	89%	48%	84%
100	51%	97%	48%	75%

From the experimentation, it is very clear that when Relu is used as an activation function at the input and hidden layers, then Sigmoid is the only preferred activation function at the output layer. In the next experimentation, Sigmoid is used as an activation function at both input and hidden layers, and all

other activation functions are used for performance evaluation. Results are tabulated in the following Table 5.

From the results, it is evident that when the Sigmoid is used as an activation function at input and hidden layers then Sigmoid only preferable at output layer also. At next phase, Softmax is used as an activation function at input and hidden layers then other activation functions are used at the output layer and performance is tabulated in the following Table 6.

Table 6. Classification with DCNN-Softmax as activation

Epochs	Relu	Sigmoid	Softmax	Tanh
10	51%	89%	48%	51%
100	51%	97%	48%	51%

From the experimentation, it is clear when Softmax is used at input and hidden layers then Sigmoid is preferable at the output layer. Finally, Tanh is used as an activation function at input and output layers. All other activation functions are tried at output layer. Performance is tabulated in the following Table 7.

Table 7. Classification with DCNN-Tanh as activation

Epochs	Relu	Sigmoid	Softmax	Tanh
10	63%	93%	48.54%	56%
100	84%	97%	48.54%	61%

From the above results, it is clearly evidencing that the Sigmoid is the most preferable activation function at the output layer when any of the activation function at input or hidden layers. If the problem is a binary class then Sigmoid is the most preferable activation function, which has yielded more accuracy in all the experiments.

5. CONCLUSION

In this work, authors have evaluated the performance of two supervised machine learning algorithms such as SVM and Deep Neural Networks on Network Intrusion Detection Systems. Now-a-days, all the services are available on internet and malicious users can attack client or server machines through this internet and avoid such attack request IDS. IDS will monitor request data, then check if it contains a normal or attack signature; if it contains the attack signature, then the request will be dropped. We have constructed Network Intrusion Detection System using SVM and DCNN and evaluated the performance using different types of kernels and different types of activation functions. The performance of the proposed method is evaluated on the NSL-KDD dataset. From the experimentation, higher accuracy achieved with DCNN compared to SVM.

REFERENCES

[1] Uppal, H.A.M., Javed, M., Arshad, M.J. (2014). An overview of intrusion detection system (IDS) along with its commonly used techniques and classifications. International Journal of Computer Science and Telecommunications, 5(2): 20-24.

[2] Chen, R.C., Chen, S.P. (2008). Intrusion detection using a hybrid support vector machine based on entropy and TF-IDF. International Journal of Innovative Computing, Information & Control, 4(2): 413-424.

[3] Hussain, A. (2018). An efficient working of network-Based Intrusion Detection System (NIDS). IJSART, 4(11): 229-232.

[4] Fernandez, G.C., Xu, S. (2019). A case study on using deep learning for network intrusion detection. 2019 IEEE Military Communications Conference, Norfolk, USA, pp. 1-6. https://doi.org/10.1109/MILCOM47813.2019.9020824

[5] Mohit, T., Raj, K., Akash, B., Jai, K. (2017). Intrusion detection system. International Journal of Technical Research and Applications, 5: 2320-8163.

[6] Mulay, S.A., Devale, P.R., Garje, G.V. (2010). Intrusion detection system using support vector machine and decision tree. International Journal of Computer Applications, 3(3): 40-43. https://doi.org/10.5120/758-993

[7] Kumar, B.S., Raju, T.C.S.P., Ratnakar, M., Baba, S.D., Sudhakar, N. (2013). Intrusion detection system-types and prevention. International Journal of Computer Science and Information Technology, 4(1): 77-82.

[8] Singh, B. (2017). A new approaches improving performance for intrusion detection system using ANN. BRDU International Journal of Multidisciplinary Research, 2(3): 41-52.

[9] Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Al-Nemrat, A., Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. IEEE Access, 7: 41525-41550. https://doi.org/10.1109/ACCESS.2019.2895334

[10] Al-Qatf, M., Lasheng, Y., Al-Habib, M., Al-Sabahi, K. (2018). Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. IEEE Access, 6: 52843-52856. https://doi.org/10.1109/ACCESS.2018.2869577

[11] Shah, B., Trivedi, B.H. (2012). Artificial neural network based intrusion detection system: A survey. International Journal of Computer Applications, 39(6): 13-18. https://doi.org/10.5120/4823-7074

[12] Sultana, N., Chilamkurti, N., Peng, W., Alhadad, R. (2018). Survey on SDN based network intrusion detection system using machine learning approaches. Peer-to-Peer Networking and Applications, 12: 493-501. https://doi.org/10.1007/s12083-017-0630-0

[13] Othman, S.M., Ba-Alwi, F.M., Alsohybe, N.T., Al-Hashida, A.Y. (2018). Intrusion detection model using machine learning algorithm on Big Data environment. Journal of Big Data, 5(1): 34. https://doi.org/10.1186/s40537-018-0145-4

[14] Hoque, M.S. (2012). An implementation of intrusion detection system using genetic algorithm. International Journal of Network Security and its Applications, 4(2): 109-120. https://doi.org/10.5121/ijnsa.2012.4208

[15] Sung, A.H., Mukkamala, S. (2003). Identifying important features for intrusion detection using support vector machines and neural networks. 2003 Symposium Applications and the Internet. Orlando, USA, pp. 209-216. https://doi.org/10.1109/SAINT.2003.1183050

[16] Yang, Y. (2013). Research on the system model of network intrusion detection. 2012 International Conference of Modern Computer Science and Applications, pp. 185-190. https://doi.org/10.1007/978-3-642-33030-8_30

[17] Niyaz, Q., Sun, W., Javaid, A.Y., Alam, M. (2015). A

deep learning approach for network intrusion detection system. EAI International Conference Bio-inspired Information and Communications Technologies. https://doi.org/10.4108/eai.3-12-2015.2262516

[18] Agarap, A.F.M. (2018). A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. ACM International Conference on Machine Learning and Computing, Macau, China, pp. 26-30. https://doi.org/10.1145/3195106.3195117

[19] Gopal, S., Sathya, M. (2018). A feature selection for intrusion detection system using a hybrid efficient model. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 3(3): 1917-1929.

[20] Stein, G., Chen, B., Wu, A.S., Hua, K.A. (2005). Decision tree classifier for network intrusion detection with GA-based feature selection. 43rd Annual Southeast Regional Conference, Kennesaw, USA, pp. 2136-2141. https://doi.org/10.1145/1167253.1167288

[21] Selvamani, D., Selvi, V. (2019). A comparative study on the feature selection techniques for intrusion detection system. Asian Journal of Computer Science and Technology, 8(1): 42-47.

[22] Anbalagan, E., Puttamadappa, C., Mohan, E., Jayaraman, B., Madane, S. (2008). Datamining and intrusion detection using back-propagation algorithm for intrusion detection. International Journal of Soft Computing, 3(4): 264-270.

[23] Khan, L., Awad, M., Thuraisingham, B. (2007). A new intrusion detection system using support vector machines and hierarchical clustering. VLDB Journal, 16(4): 507-521. https://doi.org/10.1007/s00778-006-0002-5

[24] Zhang, J., Zulkernine, M., Haque, A. (2008). Random-forests-based network intrusion. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 38(5): 649-659. https://doi.org/10.1109/TSMCC.2008.923876

[25] Mukkamala, S., Janoski, G., Sung, A. (2002). Intrusion detection using neural networks and support vector machines. Proceedings of the 2002 International Joint Conference on Neural Networks, Honolulu, USA, pp. 1702-1707. https://doi.org/10.1109/ijcnn.2002.1007774

[26] Kim, D.S., Nguyen, H.N., Park, J.S. (2005). Genetic algorithm to improve SVM based network intrusion detection system. 19th International Conference on Advanced Information Networking and Applications, Taipei, Taiwan, pp. 155-158. https://doi.org/10.1109/AINA.2005.191

[27] Vinayakumar, R., Kp, S., Poornachandran, P. (2017). Applying convolutional neural network for network intrusion detection. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, pp. 1222-1228.

[28] Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q. (2018). A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1): 41-50. https://doi.org/10.1109/TETCI.2017.2772792

[29] Mohammadpour, L., Ling, T.C., Liew, C.S., Chong, C.Y. (2018). A convolutional neural network for network intrusion detection system. The Asian-Pacific Advanced Network, 46: 50-55.

[30] Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R., Ghogho, M. (2016). Deep learning approach for network intrusion detection in software defined networking. 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, pp. 258-263. https://doi.org/10.1109/WINCOM.2016.7777224

[31] Devaraju, S., Ramakrishnan, S. (2014). Performance comparison for intrusion detection system using neural network with KDD dataset. ICTACT Journal on Soft Computing, 4(3): 743-752. https://doi.org/10.21917/ijsc.2014.0106

[32] Gurung, S., Ghose, M.K., Subedi, A. (2019) Deep learning approach on network intrusion detection system using NSL-KDD dataset. International Journal of Computer Network Information Security, 11(3): 8-14. https://doi.org/10.5815/ijcnis.2019.03.02

[33] Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A. (2015). 2015 IEEE symposium on computational intelligence for security and defense applications (CISDA). 2015 – Proceedings. 2015 IEEE Symp. Comput. Intell. Secur. Def. Appl. CISDA 2015 - Proc., no. Cisda, pp. 1–6.

[34] Popoola, E., Adewumi, A. (2017). Efficient feature selection technique for network intrusion detection system using discrete differential evolution and decision tree. International Journal of Network Security, 19(5): 660-669. https://doi.org/10.6633/IJNS.201709.19(5).02

[35] Ingre, B., Yadav, A. (2015). Performance analysis of NSL-KDD dataset using ANN. 2015 International Conference on Signal Processing and Communication Engineering Systems, Guntur, India, pp. 92-96. https://doi.org/10.1109/SPACES.2015.7058223

[36] Papamartzivanos, D., Marmol, F.G., Kambourakis, G. (2019). Introducing deep learning self-adaptive misuse network intrusion detection systems. IEEE Access, 7: 13546-13560. https://doi.org/10.1109/ACCESS.2019.2893871

[37] Fern, G.C. (2019). A case study on using deep learning for network intrusion detection. 2019 IEEE Military Communications Conference (MILCOM), Norfolk, USA, pp. 1-6. https://doi.org/10.1109/MILCOM47813.2019.9020824

[38] Peng, W., Kong, X., Peng, G., Li, X., Wang, Z. (2019). Network intrusion detection based on deep learning. 2019 International Conference on Communications, Information System and Computer Engineering (CISCE), Haikou, China, pp. 431-435. https://doi.org/10.1109/CISCE.2019.00102

[39] Abdullah, B., Abd-alghafar, I., Salama, G., Abd-alhafez, A. (2009). Performance evaluation of a genetic algorithm based approach to network intrusion detection system. Aerospace Science& Aviation Technology, 13: 1-17. https://doi.org/10.21608/asat.2009.23490

[40] Papamartzivanos, D. (2019). Introducing deep learning self-adaptive. IEEE Access, 7: 13546-13560. https://doi.org/10.1109/ACCESS.2019.2893871

[41] Kim, C., Park, J. (2019). Designing online network intrusion detection using deep. Computer & Electrical Engineering, 79: 106460. https://doi.org/10.1016/j.compeleceng.2019.106460

[42] Mukkamala, S., Sung, A.H. (2003). Feature selection for intrusion detection with neural networks and support vector machines. Transportation Research Record: Journal of the Transportation Research Board, 1822: 33-39. https://doi.org/10.3141/1822-05