Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 167 (2020) 1561–1573

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2019)

Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT

Sarika Choudhary[a,*], Nishtha Kesswani[b]

[a] *Central University of Rajasthan, Ajmer, Rajasthan, India*
[b] *Central University of Rajasthan, Ajmer, Rajasthan, India*

Abstract

Internet of Things (IoT) network is the latest technology which is used to connect all the objects near us. Implementation of IoT technology is latest and growing day-by-day, it is coming with risk itself. So, it required the most efficient model to detect malicious activities as fast as possible and accurate. In our paper, we considered Deep Neural Network (DNN) for identifying the attacks in IoT. Intelligent intrusion detection system can only be built if there is availability of an effective dara set. Performance of DNN to correctly identify the attack has been evaluated on the most used data sets, i.e., KDD-Cup'99, NSL-KDD, and UNSW-NB15. Our experimental results showed the accuracy rate of the proposed method using DNN. It showed that accuracy rate is above 90% with each dataset.

Keywords: IoT; Internet of Things (IoT); Deep Neural Network (DNN); UNSW-NB15; KDD Cup'99; NSL-KDD; Data set.

1. Introduction

Nowadays emerging technology makes one's life more comfortable, but their security and privacy are getting compromised, and, it is a significant concern factor. IoT raises lots of vulnerabilities in many terms like, network, infrastructure, things, communication etc. As there are millions of devices that is making it difficult to implement security on each and every device. To monitor the data through network can be achieved by network-based security. Network based security solutions can be implemented to IoT devices with minor changes required. To allow access over the network, IoT devices should register theirself to make sure that they are free from intruders. It is necessary to monitor the all incoming and outgoing traffic of each object. Also, maintain a template for normal behavior of the network traffic. Now, if any value fails to fall into the normal behavior category then it is identified as the attack and raised the alarm as a signal to the owner of the respected devices.

* Corresponding author.
E-mail address: scpreety98@gmail.com

10.1016/j.procs.2020.03.367

IDS (Intrusion Detection System) can be helpful to achieve the security of the network that used to observe the abnormal behavior of the network [1]. Now, main thing is that where to place an IDS in the system? If an IDS placed on the nodes or distributed randomly then it is known as Network based IDS whereas. if an IDS placed on workstations thhen it is known as Host based IDS. NIDS is more likely to used IDS. We can combine IDS with some Machine Learning Technologies So that we could get more accurate results. Machine Learning (ML) is the important part of AI that is used to analyze and construct the system based on the gained knowledge from the data sets. There are mainly three types of learning technique based on the use of labelled data, i.e., Supervised, un-supervised and, semi-supervised learning. Common machine learning algorithms are Support Vector Machine (SVM), logistic regression, naive-bayes classifier, linear regression, K-nearest neighbor (KNN), artificial neural network (ANN), deep neural network (DNN), and so on.
Deep learning (DL) is also recognized as deep structural or hierarchical learning. It is the vital algorithm of machine learning (ML) in terms of complex structure and the speed of learning data. DL requires a large amount of data, unlike ML, to find the patterns more accurate. A network designed with the help of deep learning is called Deep Neural Network (DNN). The main difference between artificial neural network and deep neural network is that if ANN contains two or more hidden layers, then it is named as deep structure. Data processing speed would be fast and learn the tasks deeper. In our paper, we are using Deep Neural Network to train, validate, and, test the network.

Rest of the paper is organized as follows: Section 2 gives the overview of Deep Neural Network and its working. Later on Section 3 gives the description of data sets features. It also explains the parameters of evaluating performance. Results and experimental details are given in section 4. At last, conclusion of our work and future scope is described in section 5.

2. Deep Neural Network

Deep learning is a feature of AI which deal with the learning approaches as human beings learn particular type of knowledge. Traditional ML approaches are linear and deep learning approaches are complex. For understanding this concept let's take an example, imagine a toddler whose first word is a cat. Toddler then learns what is a cat by seeing the object, pointing out and by saying it cat. Their parents says, "Yes, that is a cat," or, "No, that is not a cat." Toddlers continues to point the objects and become more aware with the characteristics of the cats. Now think, what the toddler does without prior knowledge. It is complicated abstraction by making a hierarchy in which each level is designed with the knowledge provided by previous layer of hierarchy (by parents for the toddler).
Deep Neural Network (DNN) is an Artificial Neural Network (ANN) including a various number of layers between the input and output layers, as shown in Figure 1. The DNN detects the accurate mathematical manipulation to make the input into the output. It would be a linear or non-linear relationship. The network moves by the layers calculating the likeliness of each output.

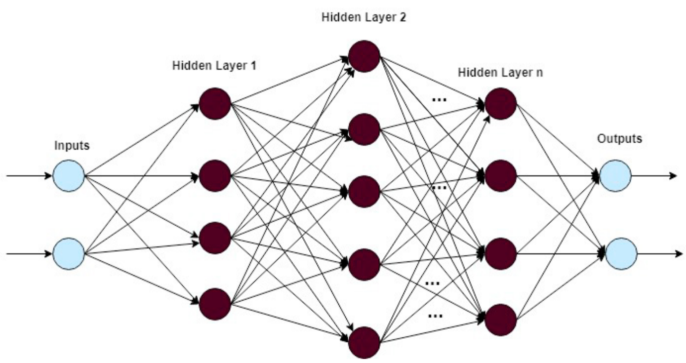


Fig. 1: Block Diagram of Deep Neural Network

Same procedure would be followed by computer programs which apply DL. Algorithms in hierarchy uses non-linear conversion on inputs and utilizes it to build a model as outputs. These repetitions lasts until output has ap-

proached to an adequate level of efficiency. Data must continue to the number of layers that is what caused the deep label.
Conventional ML process is supervised and it needs specific information by the programmer. After learning, it will tell the user for what type of image they are searching for like, whether any image contains cat or not. This process is known as feature extraction. Accurate results given by the computer are totally depends upon the ability of a programmer to explain features set for particular image. But advantage of DL is that feature set is constructed by program itself deprived of supervision. Un-supervised learning is to learn by itself. It is faster and more reliable.
At First, training data might be provided (bundle of images) in which programmer must marked each image with a label "cat" or "not cat". DL program applies this information to create a feature set for cat object and make a predictive model. This model produces forecast that anything in the picture which has "a tail" and "four legs" will be noted as "cat". It is not like program has label "four legs" and "a tail", it naturally scans the image and patterns of pixels. Predictive model become more accurate and complex after each repetitions. Deep Learning is the resemblance of human neurons, it is sometimes called DNN (Deep Neural Network) or DNL (Deep Neural Learning).
There are some types of the deep neural network, i.e., Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory (LSTM), etc. CNN is extensively used in the field of language processing [6] and image processing[10, 7]. In CNN, an image is directly fed to the model without pre-processing, then convolution operations assesses features [4]. Another type of DL method is RNN which shows promising results in the fields of text processing [3] and NLP(natural language processing) [5]. An evolution of RNN is LSTM network. LSTM is cable of learning patterns in parallel succession. It is used to classify attacked and normal data. Advantages of LSTM is that it can be directly applied to raw data without applying feature selection/extraction method.

3. Data set Description

Data sets are used to evaluate the model, whether it can be used to detect attack accurately or not. The quality of the data set eventually affects the outcome of any Network Intrusion Detection System (NIDS). Here we are considering three data sets named KDD Cup'99, NSL-KDD and, UNSW-NB15. Detail description of their features is given below.

3.1. KDD Cup'99 Data set

KDD'99 data set was created by DARPA in 1999 by using recorded network traffic from 1998 dataset. It is being pre-processed into 41 features per network connection. Features in KDD'99 data set are categorized into four groups i.e., Basic Features (#1 to #9), Content Features (#10 to #22), Time based traffic features (#23 to #31), and Host based traffic features (#32 to #41) as shown in Table 1. KDD'99 [11] consists of 4,898,430 records that is larger than other data sets. There are four main categories of attacks has shown in Table 2, these are DoS, R2L (unauthorized access from a remote machine), U2R (Unauthorized access to Root) and Probe. Many data mining techniques has been applied to the KDD'99 data set to detect intrusions in network traffic.

KDD Cup'99 is mostly used data set to build intrusion detection system (IDS). KDD data set have two critical issues concluded by the statistical analysis, that is profoundly affect the performance of the system. Most significant issue in KDD data set is that it has large number of replicated records. It is found that about 78% and 75% records are duplicate in train and test data set respectively. Huge number of replicated records may lead learning algorithms to be partial instead of numerous records. Thus, algorithm will stop learning infrequent records. These records may be harmful to network like U2R, R2L etc.

3.2. NSL-KDD Data set

To solve the issues of KDD Cup data set, they have proposed a new data set, i.e., NSL-KDD, which consists of selected records of the complete KDD Cup'99 data set. The following are the advantages of NSL-KDD data set over the KDD Cup'99 data set:

- It doesn't include irrelevant records in the train set, so the classifiers will not be partial towards more repeated records.

- From each difficulty record , number of chosen records are inversely proportional to the percentage of the records in the KDD data set. So, this results that classification percentage of the different ML (Machine Learning) techniques differ in a wide range. This make structured comprehensive evaluation of ML approaches [8].
- In train and test data set number of records are logical that make it more comfortable to do the experiments on entire data set without any need to choose random small segments. Therefore, the evaluation results of different work will be steady.

3.3. UNSW-NB15 Data set

The UNSW-NB15 dataset [9] is new and was published in 2015. It includes moderns attack (nine attack types compared to 14 attack types in KDD'99 dataset). It has 49 features and a variety of normal and attacked activities including with class labels of total 25,40,044 records. There are 2,21,876 normal records and 3,21,283 attacked records in the total number of records. Features of UNSW-NB15 data set is categorized into six groups namely Basic Features, Flow Features, Time Features, Content Features, Additional Generated Features, and Labelled Features. Features counting from 36-40 are known as General Purpose Features. Features counting from 41-47 are known as connection features. Further, UNSW-NB15 dataset has nine type of attacks category known as the Analysis, Fuzzers, Backdoors, DoS Exploits, Reconnaissance, Generic, Shellcode, and Worms. List of Features and their description has shown in Table 3

3.4. Evaluation Metrics

To increase the performance of the model; accuracy, recall, the precision rate should be calculated. We choose accuracy, recall, precision, and, F1 Measure for evaluation.
If we could create a confusion matrix, then it will be easy to calculate all the performance measures. Accuracy 1 is the percentage of true detection over total instances. Recall is how often does it predicted correct. Recall 2 is also known as True Positive Rate (TPR) or Sensitivity. Precision 3 tells that when it is predicted correct, how often is it actually correct. F1 measure 4 is a weighted average of the recall and precision. Mathematical representation of all measures can be extracted from the confusion matrix 2. In Figure 2, Actual No means actually normal records, Actual Yes means attacked records in actual, Predicted No means records that are predicted as normal and, Predicted Yes means records that are predicted as an attack. Confusion matrix is a table that is related to represent the performance of a classification model on a set of test data for which the true values are identified.

Total instances	**Predicted NO**	**Predicted YES**	
Actual NO	TN (True Negative)	FP (False Positive}	
Actual YES	FN(False Negative)	TP (True Positive)	**Recall**
		Precision	**Accuracy**

Fig. 2: Confusion matrix

$$Accuracy = \frac{TP + TN}{Totalinstances} \tag{1}$$

$$Recall = \frac{TP}{TotalActuallyYES} \tag{2}$$

$$Precision = \frac{TP}{TotalPredictedYES} \tag{3}$$

$$F1Measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

Above is the mathematical representation of performance measures.

4. Experiment and Results

We used Deep Neural Network (DNN) with 20 hidden layers for this study. We set division of the data set into training, validation, and testing. We divided it into 70, 15, 15 ratios respectively. 70% data set for training then 30% is divided equally into testing and validation.

We used an entire data set to train the network. We extracted eight attributes from the data set used as inputs and produced one output. Inputs are attributes in the data set, and output is truly detected attacked behavior. We took 8 attributes i.e #5,6,10,11,23,32,33,41 from KDD and NSL Data set. From UNSW data set we took #2,4,7,8,9,10,15,16 features for training.

Experiment work has been done in MatLab 2016b with 8GB RAM on Mac OS X. We used DNN to train and classify the network behavior. Figure 3 shows the procedure of training and testing.

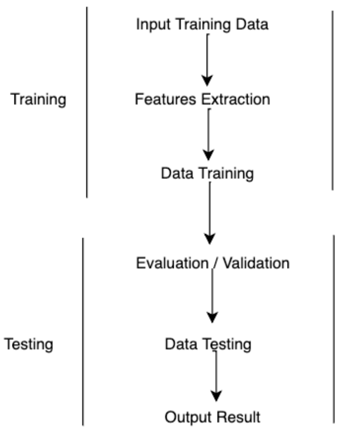


Fig. 3: Data training and testing procedure

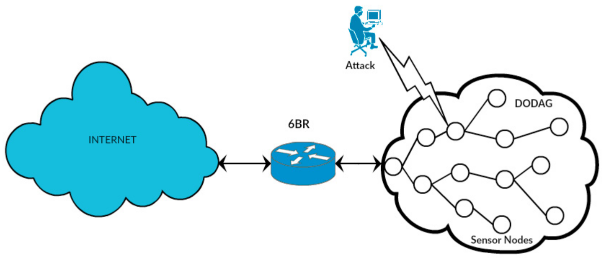


Fig. 4: Attacked IoT network architecture

As we know that, in IoT, millions and billions of devices would be there. So, we need the most reliable intrusion detection technique. A deep neural network is suitable in this scenario. It can handle complex problems and fast to learn. Hence, we are using DNN in IoT for detecting the attacks.

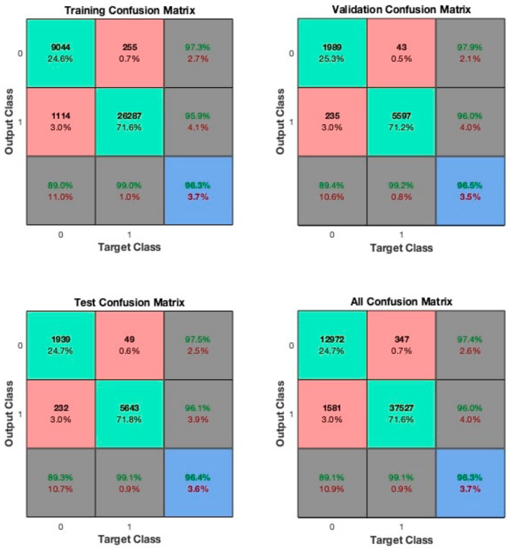
Figure 4 shows the IoT network architecture when attack is happening. In this scenario, we implemented this on 6BR (IPv6 Border Router) so that it would monitor the traffic, and based on their training, it would detect attacked behavior.

The root node is also known as 6BR. There may be on, or more 6BR in the network depends upon their network size. The root node is connected through other nodes to the child nodes. It follows the RPL (Routing Protocol for Low power and lossy network) routing technique and makes DODAG (Destination Oriented Directed Acyclic Graph) tree. In DODAG, all nodes should be connected and form a tree-like structure. In which there should be one root node, and it would be further connected to the parent's node of some child nodes and so on. There are some rules to make the DODAG in RPL. One most important rule is that the rank of nodes will increase from the root node to the child node. It should be strictly followed while making DODAG [2].

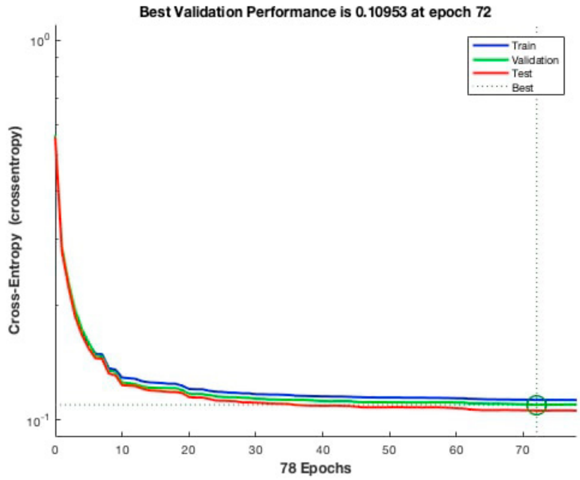
DNN based detection technique can be implemented on nodes and 6BR both. We used MatLab for implementation and check the performance of DNN. We used three data sets, namely KDD-Cup'99, NSL-KDD, and UNSW-NB15. Following are the results has shown for each data set.

Performance graph, confusion matrix, and ROC curve have displayed here to show the accuracy of DNN with each data set.

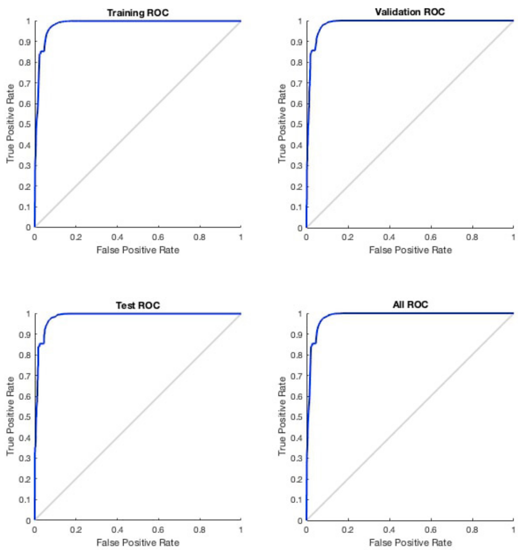
Figure 5a shows the confusion matrix for KDD dataset. It represents the accuracy of the detection that is 96.3%. It also shows the correct classification and incorrect classification. Based on these, we can calculate sensitivity, specificity, etc. Figure 5b shows the performance graph for KDD-Cup'99 data set. It is shown that DNN has still perfor-



(a)



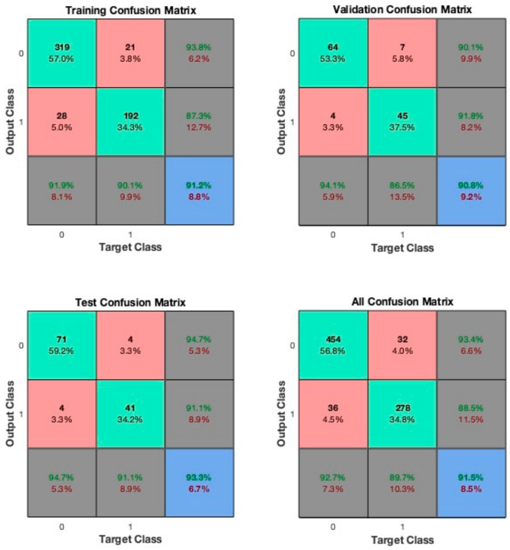
(b)



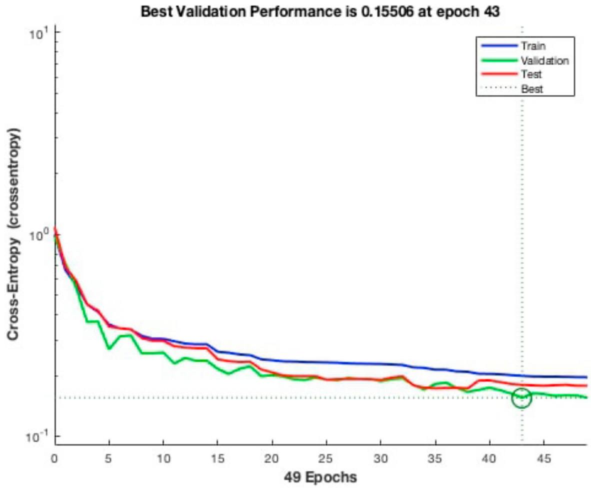
(c)

Fig. 5: (a) Deep Neural Network Confusion Matrix for KDD-Cup’99 Data set. (b) Performance using DNN for KDD-Cup’99 Data set. (c) ROC Curve for KDD-Cup’99 Data set using DNN.

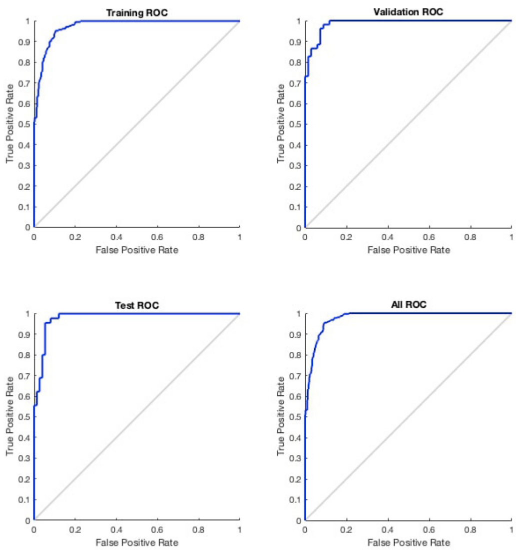
mance after 72 epochs. This graph is between epochs and cross-entropy. It is for train, validation, and test performance of the records. It gives best validation performance is 0.10953 at epoch 72. Generally, error reduces after more epochs of training, but it may start after validation, and best performance is taken from the epochs with the lowest error. Figure 5c shows the ROC (Receiver Operating Curve) curve for KDD-Cup’99 data set. It is between True Positive Rate and False Positive Rate for all; training, testing, validation. ROC curve represents the performance of detection.



(a)



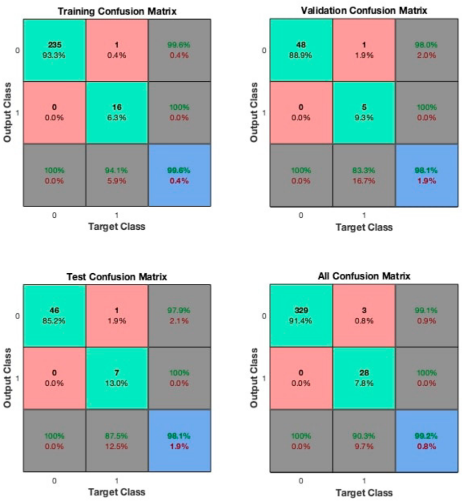
(b)



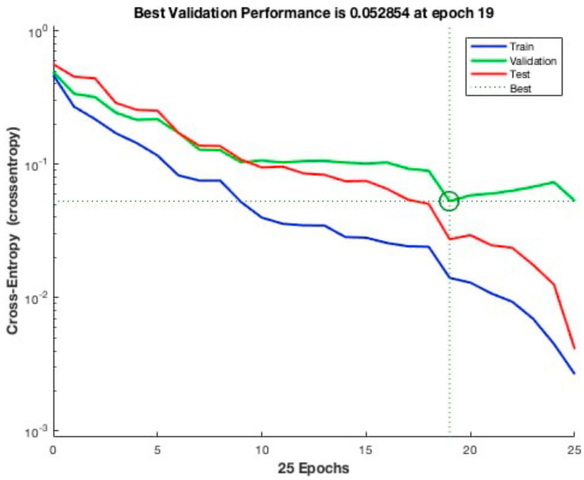
(c)

Fig. 6: (a) Confusion Matrix of NSL-KDD Data set using DNN. (b) Performance of NSL-KDD Data set using DNN. (c) ROC Curve of NSL-KDD Data set using DNN.

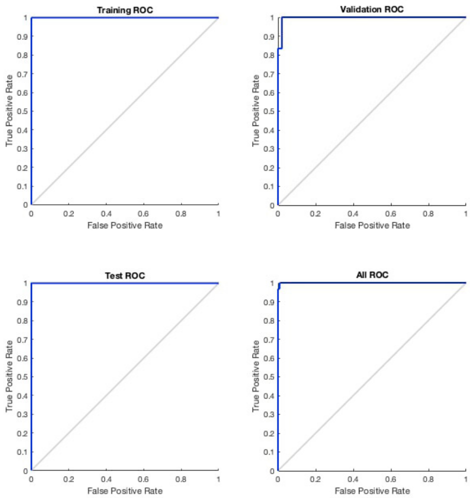
Figure 6a shows the confusion matrix of NSL data set using a deep neural network. It shows the overall accuracy of intrusion detection is 91.5%. Figure 6b represents the performance of NSL data set using a deep neural network. Best performance of the NSL data set is 0.15506 at epoch 43. ROC curve shows the graphic representation between TPR and FPR, as shown in Figure 6c for NSL data set.



(a)



(b)



(c)

Fig. 7: (a) Confusion Matrix of UNSW NB-15 Data set using DNN. (b) Performance of UNSW NB-15 Data set using DNN. (c) ROC Curve of UNSW NB-15 Data set using DNN.

Figure 7a is the confusion matrix of UNSW-NB15 data set using a deep neural network. Accuracy with data set is 99.2%. Figure 7b represents the performance graph of UNSW-NB15 data set using a deep neural network. Best validation performance is 0.052854 at epoch 19. A graph is between Cross-Entropy and epochs. Figure 7c shows the ROC curve of UNSW-NB15 data set using a deep neural network. ROC curve shows the true positive rate of detecting intrusions.

We calculated Accuracy, Recall, Precision, and F1 Measure. Figure 8 shows the overall performance of DNN with each data set. All the results show that UNSW-NB15 data set gives the best performance with DNN. It classifies more accurate among all three data sets. It is a new and updated data set for intrusion detection.

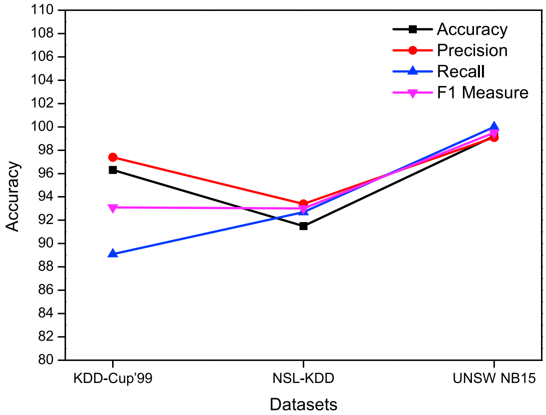


Fig. 8: Performance of DNN using Different Datasets

5. Conclusion and Future Scope

Internet of Things (IoT) is a growing technology that changed our life from good to smart. IoT devices are connected over the internet so there are more insecurities from the attacks. To find out the intrusions very effectively, researchers should build solutions. Development of smart methods are required to fight with complex new smart system. In our paper, we presented a deep neural network for intrusion detection for IoT network. Detection was based on classifying intruded patterns. We used three data sets for the train and tested our network with DNN. It has been shown that it was able to identify attacked behavior successfully. The result shows that with each data set we got at least 90% accuracy and more. Further, we will consider other deep networks like CNN (conventional Neural Network), RNN (Recurrent Neural Network), etc. for our experiment. We will also compare our technique with existing techniques.

References

[1] Choudhary, S., Kesswani, N., 2018. Detection and prevention of routing attacks in internet of things, in: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), IEEE. pp. 1537–1540.

[2] Choudhary, S., Kesswani, N., 2019. A survey: Intrusion detection techniques for internet of things. International Journal of Information Security and Privacy (IJISP) 13, 86–105.

[3] Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J., 2016. Lstm: A search space odyssey. IEEE transactions on neural networks and learning systems 28, 2222–2232.

[4] Hao, W., Bie, R., Guo, J., Meng, X., Wang, S., 2018. Optimized cnn based image recognition through target region selection. Optik 156, 772–777.

[5] Hori, T., Chen, Z., Erdogan, H., Hershey, J.R., Le Roux, J., Mitra, V., Watanabe, S., 2017. Multi-microphone speech recognition integrating beamforming, robust feature extraction, and advanced dnn/rnn backend. Computer Speech & Language 46, 401–418.

[6] Hu, B., Lu, Z., Li, H., Chen, Q., 2014. Convolutional neural network architectures for matching natural language sentences, in: Advances in neural information processing systems, pp. 2042–2050.

[7] Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, pp. 1097–1105.

[8] Meena, G., Choudhary, R.R., 2017. A review paper on ids classification using kdd 99 and nsl kdd dataset in weka, in: 2017 International Conference on Computer, Communications and Electronics (Comptelix), IEEE. pp. 553–558.

[9] Moustafa, N., Slay, J., 2015. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), in: 2015 military communications and information systems conference (MilCIS), IEEE. pp. 1–6.

[10] Sharif Razavian, A., Azizpour, H., Sullivan, J., Carlsson, S., 2014. Cnn features off-the-shelf: an astounding baseline for recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 806–813.

[11] Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A., 2009. A detailed analysis of the kdd cup 99 data set, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, IEEE. pp. 1–6.

Table 1: KDD Cup'99 Data set Features List with Description

Attribute Number	Features	Description
1	duration	Length of the time duration of the connection
2	protocol_type	Protocol used
3	service	Service used by destination network
4	flag	Status of the connection (Error or Normal)
5	src_bytes	Number of data bytes transferred from source to destination
6	dst_bytes	Number of data bytes transferred from destination to source
7	land	If source and destination port no. and IP addresses are same then it will set as 1 otherwise 0
8	wrong_fragment	Total number of wrong fragments in a connection
9	urgent	Number of urgent packets (these packets with urgent bit activated)
10	hot	Number of 'hot' indicators means entering in a system directory
11	num_failed_logins	Number of failed login attempts
12	logged_in	Shows login status (1- successful login, 0- otherwise)
13	num_compromised	Number of compromised conditions
14	root_shell	Shows root shell status (1-if root shell obtained otherwise 0)
15	su_attempted	Set as 1 if 'su_root' command used otherwise set as 0
16	num_root	Number of operations performed as root
17	num_file_creations	Number of file creation operations
18	num_shells	Number of shell prompts in a connection
19	num_access_files	Number of operations on access control files
20	num_outbound_cmds	Number of outbound commands in a ftp session
21	is_host_login	If login as root or admin then this set as 1 otherwise 0
22	is_guest_login	Set as 1 if login as guest otherwise 0
23	count	Number of connections to the same destination host
24	srv_count	Number of connection to the same service (port number)
25	serror_rate	Percentage of connections that have activated flag (#4) s0,s1,s2 or s3, among the connections aggregated in count (#23)
26	srv_serror_rate	Percentage of connection that have activated flag (#4) s0,s1,s2 or s3, among the connections aggregated in srv_count (#24)
27	rerror_rate	Percentage of connections that have activated flag (#4) REJ, among the connections aggregated in count (#23)
28	srv_rerror_rate	Percentage of connections that have activated flag (#4) REJ, among the connections aggregated in srv_count (#24)
29	same_srv_rate	Percentage of connections that were to the same services, among the connections aggregated in count (#23)
30	diff_srv_rate	Percentage of connections that were to the different services, among the connections aggregated in count (#23)
31	srv_diff_host_rate	Percentage of connections that were to different destination machines among the connections aggregated in srv_count (#24)
32	dst_host_count	Number of connections having the same destination host IP address
33	dst_host_srv_count	Number of connections having same port number
34	dst_host_same_srv_rate	Percentage of connections that were to the same service among the connections aggregated in dst_host_count (#32)
35	dst_host_diff_srv_rate	Percentage of connections that were to different service among the connections aggregated in dst_host_count (#32)
36	dst_host_same_src_port_rate	Percentage of connections that were to the same source port among the connections aggregated in dst_host_srv_count (#33)
37	dst_host_srv_diff_host_rate	Percentage of connections that were to the different destination machines among the connections aggregated in

Attribute Number	Feature	Description
38	dst_host_serror_rate	Percentage of connections that have activated flag (#4) s0,s1,s2 or s3, among the connections aggregated in dst_host_count (#32)
39	dst_host_srv_serror_rate	Percentage of connections that have activated flag (#4) s0,s1,s2 or s3, among the connections aggregated in dst_host_srv_count (#33)
40	dst_host_rerror_rate	Percentage of connections that have activated flag (#4) REJ, among the connections aggregated in dst_host_count (#32)
41	dst_host_srv_rerror_rate	Percentage of connections that have activated flag (#4) REJ, among the connections aggregated in dst_host_srv_count (#32)
42	label	Attack class label

Table 2: Attacks classification in KDD Cup'99 data set

Attack Category	Attack Type
DoS (Denial of Service)	Neptune, land, pod, smurf, teardrop, back, worm, udpstorm, processtable, apache2 (10)
Probe	ipsweep, satan, nmap, portsweep, mscan, saint (6)
R2L	ftp_write, guess_password, imap, multihop, phf, spy, warezclient, warexmaster, snmpguess, named, xlock, xsnoop, snmpgetattack, httptunnel, sendmail (15)
U2R	buffer_overflow, loadmodule, perl, rootkit, ps, xterm, sqlattack (7)

Table 3: Features List of UNSW-NB15 Data set with Description

Attribute Number	Feature	Description
1	srcip	Source IP address
2	sport	Source port number
3	dstip	Destination IP address
4	dsport	Destination port number
5	proto	Transaction protocol
6	state	Indicates to the state and its dependent protocol, e.g. ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, and (-) (if not used state)
7	dur	Record total duration
8	sbytes	Source to destination transaction bytes
9	dbytes	Destination to source transaction bytes
10	sttl	Source to destination time to live value
11	dttl	Destination to source time to live value
12	sloss	Source packets retransmitted or dropped
13	dloss	Destination packets retransmitted or dropped
14	service	http, ftp, smtp, ssh, dns, ftp-data ,irc and () if not much used service
15	Sload	Source bits per second
16	Dload	Destination bits per second
17	Spkts	Source to destination packet count
18	Dpkts	Destination to source packet count
19	swin	Source TCP window advertisement value
20	dwin	Destination TCP window advertisement value
21	stcpb	Source TCP base sequence number
22	dtcpb	Destination TCP base sequence number
23	smeansz	Mean of the packet size transmitted by the source
24	dmeansz	Mean of the packet size transmitted by the destination
25	trans_depth	Represents the pipelined depth into the connection of http request/response transaction
26	res_bdy_len	Actual uncompressed content size of the data transferred from the server's http service
27	Sjit	Source jitter (mSec)
28	Djit	Destination jitter (mSec)
29	Stime	record start time
30	Ltime	record last time
31	Sintpkt	Source interpacket arrival time (mSec)
32	Dintpkt	Destination interpacket arrival time (mSec)
33	tcprtt	TCP connection setup round-trip time, the sum of 'synack' and 'ackdat'.
34	synack	TCP connection setup time, the time between the SYN and the SYN_ACK packets.
35	ackdat	TCP connection setup time, the time between the SYN_ACK and the ACK packets.
36	is_sm_ips_ports	If source (#1) and destination (#3) IP addresses equal and port numbers (#2)(#4) equal then, this variable takes value 1 else 0
37	ct_state_ttl	No. for each state (#6) according to specific range of values for source/destination time to live (#10) (#11).
38	ct_flw_http_mthd	No. of flows that has methods such as Get and Post in http service.
39	is_ftp_login	If the ftp session is accessed by user and password then 1 else 0.
40	ct_ftp_cmd	No of flows that has a command in ftp session.
41	ct_srv_src	No. of connections that contain the same service (#14) and source address (#1) in 100 connections according to the last time (#26).

Attribute Number	Feature	Description
42	ct_srv_dst	No. of connections that contain the same service (#14) and destination address (#3) in 100 connections according to the last time (#26).
43	ct_dst_ltm	No. of connections of the same destination address (#3) in 100 connections according to the last time (#26).
44	ct_src_ ltm	No. of connections of the same source address (#1) in 100 connections according to the last time (#26).
45	ct_src_dport_ltm	No of connections of the same source address (#1) and the destination port (#4) in 100 connections according to the last time (#26).
46	ct_dst_sport_ltm	No of connections of the same destination address (#3) and the source port (#2) in 100 connections according to the last time (#26).
47	ct_dst_src_ltm	No of connections of the same source (#1) and the destination (#3) address in in 100 connections according to the last time (#26).
48	attack_cat	The name of each attack category.
49	Label	0 for normal and 1 for attack records