DEEP LEARNING APPROACH FOR SECURING DATA IN CLOUD AGAINST DDOS ATTACKS

Meenakshi[1], Ramachandra AC[2]
[1]Research Scholar, Computer Science & Engineering Department,
Nitte Meenakshi Institute of Technology, Bangalore, India-560064.
meenakshi.rao.kateel@gmail.com
[2]Professor & Head, Electronics & Communication Engineering Department,
Nitte Meenakshi Institute of Technology, Bangalore, India- 560064.
ramachandra.ac@nmit.ac.in

ABSRACT: DDoS attack is launched remotely against public servers and effects legitimate network users. It is one of the greatest cyber threats to the availability of networks, applications, and services. Gaming, attack capability shows, fun, extortion, creating huge loss to business etc. are some of the top motivations for the criminals behind these attacks. Exploring DDoS attack detection and classifying attack types were main objective of this paper. To achieve this Deep Neural Network model is designed. Model is trained and tested using dataset CICDDoS-2019 which is appropriate to gain information about most recent DDoS attack types. Feature Engineering is done on dataset samples to make it fit for modelling and evaluation. Proposed model classifies packets as benign or attacked. Accuracy, precision, F1-measure and recall are some of the metrics considered here for performance measurement. Results show that proposed classification model performs well, as compare to the traditional machine learning classifiers. This study result contributes to a good understanding of IDS capacity using open-source DDoS dataset.

KEY WORDS: DDoS attacks, Deep Neural Networks, Feature selection, Pre-processing

1. INTRODUCTION

Gligor framed the new term “Denial of Service”, (DoS) in the context of Operating Systems [1, 2]. Secured networking infrastructure also, mostly suffers due to *bot* and DDoS attacks which are usually difficult to be detected as doubtful. It is due to attack is aimed on resource allocation system and could be seen as heavy use of resources which is common some times. By seeing large number of SYN, ICMP request packets, firewalls may get suspicious but cannot come to any conclusion. Attackers’ request triggers the response patterns to be generated from the targeted side which may consume device’s power, CPU utilization, memory, RAM etc. anything. Data stealing is not intention behind attack; instead interrupt network services to cause the organization experience great loss.

1.1 DDoS Attack Classifications

Attack type	TCP based	UDP based	TCP/UDP based
Reflection Attacks	MSSQL, SSDP	CharGen, NTP, TFTP	DNS, LDAP, NetBios, SNMP, Portmap
Exploitation attacks	SYN flood	UDP flood, UDP-lag	

Table 1.1 DDoS attack classification

In Reflection attacks attacker identity is hidden by some other genuine IP address. This phenomenon is known as IP spoofing. IP address of target victim is used as source address. After attack, target victim will start receiving response packets in large quantity. Exploitation attacks are flaw exploitation type where the “target is the software of the system, attempting to deplete its resources like memory, CPU, disk space or memory buffers”. In flooding attacks, the attack is on the “networking capabilities of the target, depleting the network capacity by accessing the resource with the means of attack”, thus making it inaccessible for legitimate users.

“Distributed” in DDoS term means, attack is not from single place. It is from the multiple systems or multiple services or multiple bots. Attack is distributed here. “Denial of Service” means service is not available or excluded. Attackers generate a large volume of packets or requests by using compromised or controlled sources to generate the attack. These requests overwhelm the target system, due to which it performs poorly and becomes unavailable to legitimate users. Intension is to bring the target system down by either congesting its networks or by depleting its resources. Depleting server resources is done by making it super busy by sending fake requests, fake packets. So that server will be busy with processing these fake one’s and depletes all its own resources, for generating fake connections and connections which never be closed. Real users will never be able to connect to server. DDoS attack can be segregated based on which layer of Open System Interconnection (OSI) model they attack. Most common attacks are at the layers Application (L7), Presentation (L6), Transport (L4) and Network (L3). With reference to OSI layers, we have segregated DDoS attacks. It is shown in Table 2.2. Most common attacks are happening in Application layer, Presentation layer, Transport layer and Network layer of OSI.

Layer	**OSI Layer**	**Data form**	**Functionality**	**Attack Types**
7	Application Layer	Data	Network Process to Application	HTTP Floods, DNS query floods, Cache-busting attacks
6	Presentation Layer	Data	Data representation, Encryption	SSL Abuse
5	Session Layer	Data	Inter-host communication	N/A
4	Transport Layer	Segments	End-to-end connection, reliability	SYN Floods
3	Network Layer	Packets	Path Determination, Logical addressing	UDP Reflection Attacks
2	Data-Link Layer	Frames	Physical Addressing	N/A
1	Physical Layer	Bits (raw data)	Media, Signal Binary Transmission	N/A

Table 1.2 DDoS attacks w.r.t OSI Model [3]

In application layer, attacker targets to application itself. "In these attacks, similar to SYN flood infrastructure attacks, the attacker attempts to overload specific functions of an application to make the application unavailable or unresponsive to legitimate users. Sometimes this can be achieved with very low request volumes that generate only a small volume of network traffic. This can make the attack difficult to detect and mitigate". Just like SYN flood attack, "attacker attempts to overload specific functions of an application", and hence it will become "unresponsive or unavailable" to true users.

HTTP Flood Attack at **Application Layer**- Let us consider HTTP webserver is under DDoS attack. Attacker (hacker) decides to bring the web server down. Attacker sets up multiple servers or applications. These can be machines or bots. Generally, bots are used now-a-days. Bots may be in the range of 100's or thousands or even billions. All these N number of bots start sending either GET / POST requests to HTTP webserver. All these fake requests go to server simultaneously, hitting the server at the same time. Server will be overwhelmed and will not understand how to handle millions of such requests simultaneously. Resources will not be enough to satisfy all. True users will no longer be able to reach webserver. They get message saying "server is busy" or something similar. They are denied by service. Here distributed attack is flooded towards HTTP web server. Service goes down.

DNS query attack in **Application layer** – It works in similar way as that of HTTP flood attack. In place of webserver, consider there is a **DNS server**. Suppose one legitimate user wants to go to xyz.com website. For his request, DNS sends back its IP address say 172. 190.1.1. Assume it is under attack by fake request similar to that of HTTP server attack mentioned earlier. Say hundreds or thousands of fake requests are coming to DNS server for seeking IP address of various websites. DNS starts responding and starts sending IP addresses. Legitimate user gets message saying "server not found or busy".

Cache-busting attacks- these are another type of "HTTP flood". Content Delivery Network CDN's job is to search requested content in cache and return. But here attacker used different query to skip CDN caching. And hence CDN need to get the thigs from "original server for every page request". It is real burden for application web server.

SSL abuse attack at **Presentation Layer** – Let us consider of having **HTTPs server** and users are trying to connect to **SSL port** for establishing connection to perform SSL handshake. Due to attackers' fake requests; it causes opening lots of connections but not closing or closing after some period. In another (second) approach, attackers' request may be trying to connect to SSL port and trying to negotiate SSL handshake. Just intension is to keep that port busy. Sometimes it may try for negotiating SSL encryption. In one more possibility (third case) hacker bots may bombard or send lots of garbage data packets to SSL port. As a result, SSL port will be engaged with listening and accepting those dummy packets, opening those packets to see what should be done. Due to this flood, SSL port will be busy and not available to true users.

SYN flood at Transport Layer- "When a user connects to a Transmission Control Protocol (TCP) service, such as a web server, their client sends a SYN synchronization packet". In another way, when a laptop user or independent user sends SYN (SYnchronize sequence

Number) request to server in order to establish connection, server sends back SYN ACK acknowledgement response. Receiver sends back another ACK acknowledgement message as a response to say that "okay I have received your response". After this 3 step handshake signalling, server closes this particular connection establishment process.

But when bots are involved, they never complete the last step of sending ACK back to server. Hence communication is incomplete. Due to this, server will have lot of open connections, it causes depleting its resources due to responding for SYN request but waiting for completion. Genuine request suffers from lack of responding from server.

UDP reflection attacks at Network layer- "User Datagram Protocol (UDP) reflection attacks exploit the fact that UDP is a stateless protocol". UDP compatible attacker system, prepares a "request packet with spoofed source IP address" as that of target. That is, source IP address of request packet is now 198.51.1.4. It does trick now, by sending this to intermediate UDP server. Let us see how target server will be under attack then.

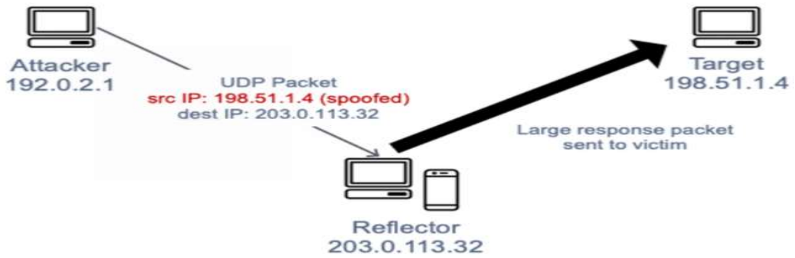


Fig. 2.1 UDP Reflection Attack at Network Layer

Intermediate UDP server responds back with high volume packet. Here intermediate server is used to amplify the attack traffic. Attacker can send requests to couple of such UDP server (amplifiers). Server is overwhelmed by so many requests, that too with high volume responses. It is under attack now. Amplification factor varies from protocol to protocol.

Signature based and anomaly-based Machine learning techniques [4,5] are some of the approaches to address DDoS attacks. Deep learning methods for KDD and other datasets are discussed in [6-15]. "Deep learning is a collection of statistical techniques of ML to learn feature hierarchies that are actually based on artificial neural networks". In 1957 the first neural network was designed by Frank Rosenblatt. NN (Nearest Neighbour) algorithm was written in 1967. Geoffrey Hinton coined the term Deep Learning. This term explains the set of algorithms used by computer to recognize and distinguish objects and texts within images and videos. Facebook's "DeepFace" is an application used to recognize individuals from photos.

1.2 Types of Neural Networks

DNN (Dense Neural Network) - Dense multiple layers are present in this network. Every neuron of one layer is connected to every neuron of other layer. One layer is connected to another layer, except the last one. Number of features and number of input layer's neurons are same. Output layer has one output if it is binary classifier, or else it depends on how many label classes are present.

Autoencoder – It is one type of DNN but not for classification, instead it is for compressing input features into lower dimension. Encoder and decoder are two parts of autoencoder. *Encoder* reduces input feature dimension and *decoder* reconstructs back to normal. And hence input and output should be same. If so then only those compressed features is ready to use for classification, because they represent original features.

CNN Convolutional Neural Network- This neural network manages task related to images. Convolutional layer filters local features and passes to next layer. It is mainly to process images; still some researchers are working on NIDS.

RNN Recurrent Neural Network - Time series establishment tasks such as NLP uses RNN. RNN has special memory storage *gates* to keep track of earlier fed features for certain amount of time. IDS system can also be built using this mechanism.

2. METHODS

We selected CICDDoS-2019 dataset which is recent one which reflects all new types of attacks. We came up with design of prediction model and it is presented in Figure 2.1. Table 1.1 shows system requirements for implementing prediction model. Table 2.2 gives all features list present in the dataset.

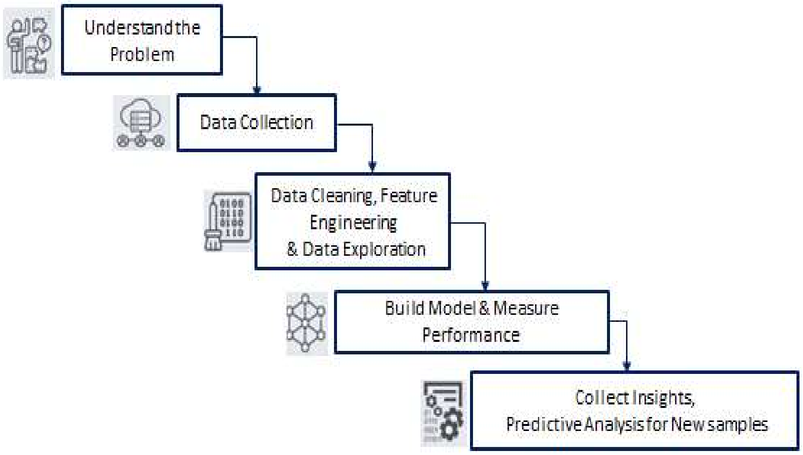

Understand the Problem
Data Collection
Data Cleaning, Feature Engineering & Data Exploration
Build Model & Measure Performance
Collect Insights, Predictive Analysis for New samples

Fig.2.1: Model Framework

Parameter	Specification details
Manufacturer	Lenova
Version	10.0.19042 Build 19042
System Type	64-bit OS, x64-based processor
OS	Windows 10 Pro
Processor Type	Intel Core Pro i5-3230M
Processor Speed	2.6GHz
Installed Memory RAM	12GB
Machine Learning Tool	Anaconda Navigator (4.10.1), Spyder (3.8.10), Jupyter Notebook
Programming Language	Python
Libraries/Packages	• tensorflow v2.3.0 • keras v2.4.0 • scikit-learn v0.24.2 (model building tool)

	• pandas v1.2.4 (data loading, cleaning tool) • matplotlib v3.3.4 (data visualization tool) • numpy v1.20.3 (data cleaning tool)

Table 2.1: System Configuration

Details of some of the important features play significant role in classifying packet as “attack” are given here.

“packet Length Std” - Usually botnets and automated tools are used to conduct DDoS attacks. Hence packet size will be of similar and fixed-size. Contrast to that, benign packets are always of varying size. In this way “*packet Length Std”* is one of the features playing significant role while determining packet is *benign* or not.

“ACK Flag Count” and “***Flow Duration”*** - In ***Syn*** attack type, server will not get respond packet from source. Attacker is playing role of source. He never sends the ACK back to server. Server keeps waiting. And hence *ACK* code will not be updated accordingly. It is exploitation of “TCP protocol weakness”.

“Protocol” and “Fwd Packets/s” - These features together determine MSSQL attack which is one of the reflection-based attack. “*Microsoft SQL Server Resolution”* Protocol (MC-SQLR) of Microsoft SQL server listens on UDP port 1434. This resolution protocol is required to address the client query to MSSQL server. Server response will be list of database instances and assistance to “which database instances they are attempting to establish connection with”. Hackers can take control over SQL server by running request scripts using a “forged IP address” to have appearance like; request is coming from the target server. “The number of existing instances present on the affected SQL server determines the power or amplification factor of the attack.”

#	Feature name	#	Feature name	#	Feature name
1	“Flow ID”	31	“Fwd IAT Min”	61	“Avg Bwd Segment Size”
2	“Source IP”	32	“Bwd IAT Total”	62	“Fwd Header Length.1”
3	“Source Port”	33	“Bwd IAT Mean”	63	“Fwd Avg Bytes/Bulk”
4	“Destination IP”	34	“Bwd IAT Std”	64	“Fwd Avg Packets/Bulk”
5	“ Destination Port”	35	“Bwd IAT Max”	65	“Fwd Avg Bulk Rate”
6	“Protocol”	36	“Bwd IAT Min”	66	“Bwd Avg Bytes/Bulk”
7	“Timestamp”	37	“Fwd PSH Flags”	67	“Bwd Avg Packets/Bulk”
8	“Flow Duration”	38	“Bwd PSH Flags”	68	“Bwd Avg Bulk Rate”
9	“Total Fwd Packets”	3	“Fwd URG Flags”	6	“Subflow Fwd

		9		9	Packets”
10	“Total Backward Packets”	40	“Bwd URG Flags”	70	“Subflow Fwd Bytes”
11	“Total Length of Fwd Packets”	41	“Fwd Header Length”	71	“Subflow Bwd Packets”
12	“Total Length of Bwd Packets”	42	“Bwd Header Length”	72	“Subflow Bwd Bytes”
13	“Fwd Packet Length Max”	43	“Fwd Packets/s”	73	“Init_Win_bytes_forward”
14	“Fwd Packet Length Min”	44	“Bwd Packets/s”	74	“Init_Win_bytes_backward”
15	“Fwd Packet Length Mean”	45	“Min Packet Length”	75	“act_data_pkt_fwd”
16	“Fwd Packet Length Std”	46	“Max Packet Length”	76	“min_seg_size_forward”
17	“Bwd Packet Length Max”	47	“Packet Length Mean”	77	“Active Mean”
18	“Bwd Packet Length Min”	48	“Packet Length Std”	78	“Active Std”
19	“Bwd Packet Length Mean”	49	“Packet Length Variance”	79	“Active Max”
20	“Bwd Packet Length Std”	50	“FIN Flag Count”	80	“Active Min”
21	“Flow Bytes/s”	51	“SYN Flag Count”	81	“Idle Mean”
22	“Flow Packets/s”	52	“RST Flag Count”	82	“Idle Std”
23	“Flow IAT Mean”	53	“PSH Flag Count”	83	“Idle Max”
24	“Flow IAT Std”	54	“ACK Flag Count”	84	“Idle Min”
25	“Flow IAT Max”	55	“URG Flag Count”	85	“SimillarHTTP”
26	“Flow IAT Min”	56	“CWE Flag Count”	86	“Inbound”
27	“Fwd IAT Total”	57	“ECE Flag Count”	87	“Label”
28	“Fwd IAT Mean”	58	“Down/Up Ratio”		
2	“Fwd IAT Std”	5	“Average Packet		

9		9	Size”
3	“Fwd IAT Max”	6	“Avg Fwd Segment
0		0	Size”

Table2.2 Features of CICDDoS2019 Dataset

***“IAT”* related features** – TFTP, UDP-lag, NTP type attacks are caught by analysing IAT features. “Bursty behaviour” of sending packets in DDoS attacks affects arrival rate and hence IAT related features.

“min_seg_size_forward” – TCP after accepting data, it breaks it and adds header to make it TCP segment. Source machine of attacker need to send more packets to victim beyond the capacity of victim can handle. Attacker uses SYN or ICMP packets, similar but smaller in order to manage computing resource at low cost. Hence packets’ “minimum segment size” malicious flow is lesser than of benign. In Data pre-processing stage we rectified missing values, categorical values, inter-dependent features, unnecessary features and applied quantile transform to bring data to proper distribution. Table2.3 gives an algorithm to pre-process the data. After applying feature selection method RFE, dataset has reduced to 22 important features.

***Algorithm*: data_clean_explore**

```
Step1□Create Data-frame from CSV/Excel/json
Step2□ Reduce memory usage of the data-frame
Step3□ Handle missing data, fillna, dropna
Step4□Handle infinity value by dropping or filling
Step5□ Encode Categorical data by label or one-hot encoding
Step6□Merge, Concatenate Data-frames
Step7□ Feature Elimination, Data transformation
Step8□ Explore, Visualize data samples by drawing charts
```

Table 2.3 Algorithm to pre-process data

2.1 Different stages in Building Predictive DNN Model

Load Dataset

- Define required functions, classes
- Load dataset
- Visualize dataset using plotting charts like histogram, scatter plot
- Check the correlation plots (with *seaborn*)
- Pre-process the data
- Decide dependent(o/p) and independent(i/p) features
- Split data columns into input(X) and output(y) features
- Divide dataset into train set and test set

Define Model

- Specify #of i/p features using *input_dim*
- Specify #of hidden layers, neurons in each layer, input/output size

- Define activation function for each layer

Compile the Model

- It uses libraries of backend TensorFlow/Theano
- Define loss function using *loss* argument
- Define optimizer with learning rate
- Define metrics to report accuracy using *metrics* argument

Fit model/ Train

- Define # of iterations using *epochs argument*
- Specify #of rows to be considered before the model weights are updated with in each epoch
- Set it using the *batch_size* argument.

Evaluate Model

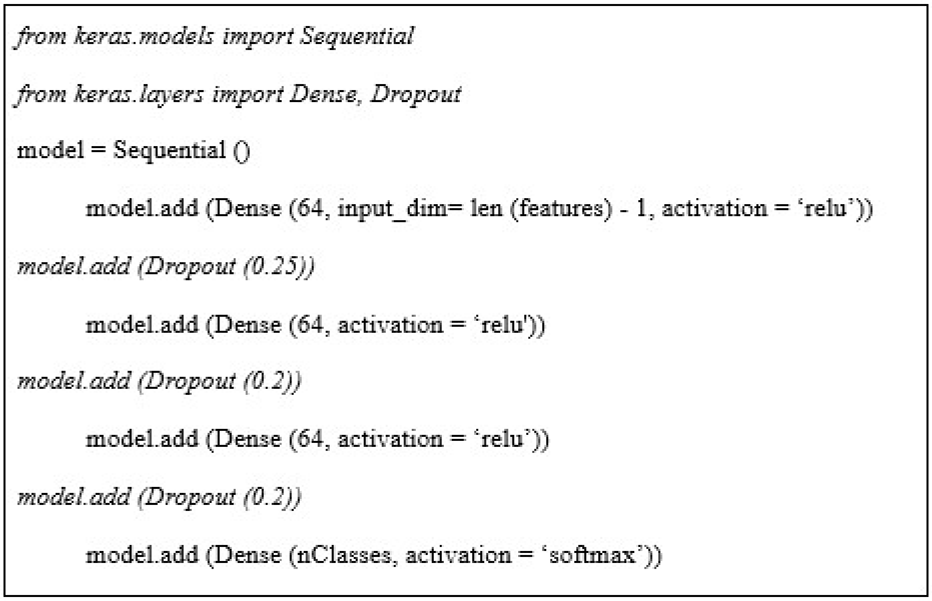
- Generate average loss and metrics (accuracy here)
- Draw charts for better visualization

Make Predictions

- Apply predict () or predict_classes() for predictions

3. RESULTS

Due to many advantages of having less but most effective list of features in dataset, major focus is given for feature selection and dimensionality reduction techniques.



```
from keras.models import Sequential

from keras.layers import Dense, Dropout

model = Sequential ()

        model.add (Dense (64, input_dim= len (features) - 1, activation = ‘relu’))

model.add (Dropout (0.25))

        model.add (Dense (64, activation = ‘relu'))

model.add (Dropout (0.2))

        model.add (Dense (64, activation = ‘relu’))

model.add (Dropout (0.2))

        model.add (Dense (nClasses, activation = ‘softmax’))
```

Fig.3.1. Code snippet of Neural Network Design

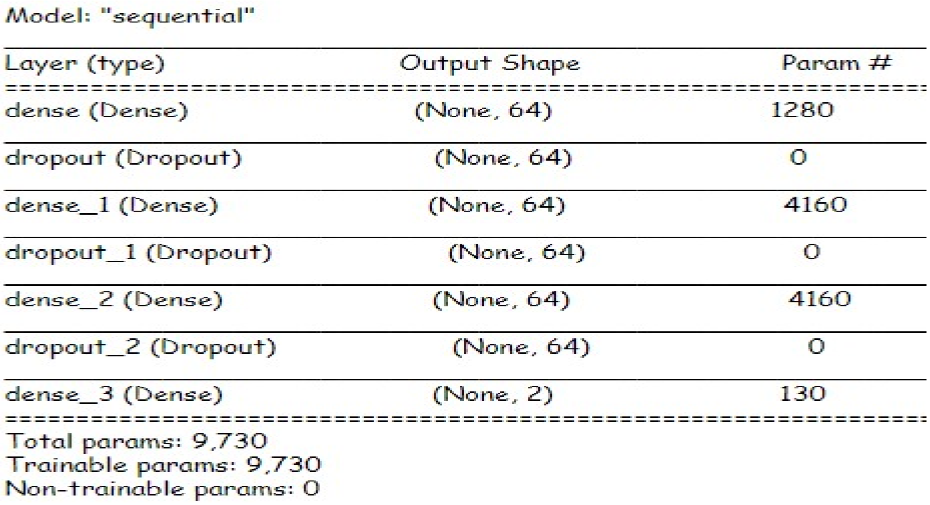
#	Features
1	“ID”
2	“Total Fwd Packets”
3	“Total Backward Packets”
4	“Total Length of Bwd

	Packets”
5	“Fwd Packet Length Max”
6	“Fwd Packet Length Min”
7	“Flow IAT Mean”
8	“Bwd IAT Total”
9	“Bwd IAT Max”
10	“Fwd Header Length”
11	“Bwd Packets/s”
12	“Min Packet Length”
13	“Max Packet Length”
14	“Packet Length Mean”
15	“ Length Std”
16	“Down/Up Ratio”
17	“Average Packet Size”
18	“Subflow Fwd Packets”
19	“Subflow Fwd Bytes”
20	“Subflow Bwd Bytes”
21	“Init_Win_bytes_forward”
22	“Idle Min”

Table 3.1: Selected features from RFE

We have implemented one of the most effective methods that is Random Forest algorithm and result obtained is given in Table 3.1.

While building model, dropout () function is required to overcome overfitting problem that may arise due to interdependency of features. Code and output model is given in Fig.3.1 and 3.2 respectively.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                1280
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 130
=================================================================
Total params: 9,730
Trainable params: 9,730
Non-trainable params: 0
```

Fig.3.2. Output Model

Results are derived from confusion matrix obtained and presented in Table 3.2.

#	**Parameter**	**Percentage**

1	Accuracy	0.9981
2	F1-score	0.9990
3	Recall	0.9999
4	Precision	0.9980

Table3.2 Results

4. CONCLUSION

Many of the research papers show that model is developed using old datasets which will not reflect current changes. Many datasets are outdated now. They all may not cover new varieties of attacks. We have taken recent dataset CICDDoS 2019 and analysed properties of that dataset, applied data cleaning methods to the dataset. We designed a DDoS attack detection framework based on Deep Learning. We applied RFE method to extract most useful features. Confusion matrix is derived and it shows 99% accuracy for binary classification. Future work will be multiclass classification to classify different types of DDoS attacks.

5. REFERENCES

[1] K. Lakshminarayanan, D. Adkins, A. Perrig, and I. Stoica, “Taming IP packet flooding attacks,” ACM SIGCOMM Computer Communication Review, vol. 34, no. 1, pp. 45–50, 2004.

[2] V. D. Gligor, “A note on denial-of-service in operating systems,” IEEE Transactions on Software Engineering, no. 3, pp. 320–324, 1984.

[3] Namratah Shah, “DDoS Attack and Types- Simplified”, 2020. [Online]. Available: https://www.youtube.com/watch?v=Wan1S9RJacU, [Accessed on 13-August- 2020].

[4] Marwane Zekri, Said El Kafhali, Noureddine Aboutabit and Youssef Saadi , “DDoS attack detection using machine learning techniques in cloud computing environments”, International Conference of Cloud Computing Technologies and Applications, 2017.

[5] Bhuyan, M. H., Kashyap, H. J., Bhattacharyya, D. K., & Kalita, J. K, “Detecting distributed denial of service attacks: Methods, tools and future directions”, The Computer Journal, vol. 4, pp. 537–556, 2014.

[6] A. Saired, R.E. Overill and T. Tadzik, “Artificial Neural Networks in the Detection of Known and Unknown DDoS attacks:Proof-of-Concept”, Commun. Coput. Inf. Sci., vol. 430, pp. 300-320, 2014.

[7] T. Thapngam, S. Yu, W. Zhou and S.K. Makki, “Distributed Denial of Service (DDoS) detection by traffic pattern analysis”, Peer-toPeer netw. Appl., vol. 7, no. 4, pp. 346-358, 2014.

[8] R. Karimazad and A. Faraahi, “An anomaly-based methods for DDoS attacks detection using RBF Neural networks”, in International Conference on network and Electronics Engineering, 2011, vol. 11, pp. 44-48.

[9] M. kale, “DDOS Attack Detection based on Ensemble of neural classifier”, Computer Science network Security, vol. 14, no. 7, pp. 122-129, 2014.

[10] M. Alenezi and M. Reed, “Methodologies for detecting DoS/DDoS attacks against network servers”, Conference on Systems and networks, 2012, pp. 92-98.

[11] G. Preetha, BSK. Devi and S.M. Shalinie, “Autonomous agent for DDoS attack detection and defense in an experimental testbed”, Fuzzy Systems, vol. 16, no. 4, pp. 520-528, 2014.

[12] Gragan Perakovic, Marko Perisa, Ivan Cvitic and Sinisa Husnjak, “Model for Detection and Classification of DDoS traffic based on ANN”, Telfor Journal, vol. 9, no. 1, pp. 26-31, 2017.

[13] CAIDA, “CAIDA: the Cooperative Association for Internet Data Analysis”, 2008. [Online]. Available: http://www.caida.org/. [Accessed on 01-Sept 2020].

[14] S.C. of Excellence, “UNB ISCX Intrusion Detection Evaluation Dataset”, 2010. [Online]. Available: http://www.unb.ca/research/iscx/dataset/iscx-IDS-dataset.html. [Accessed on 01-Sept- 2020].

[15] J.J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, Z. Granville, and A.L. Pras, “Booters- An analysis of DDoS-as-a-service attacks”, Integrated network management, 2015, pp. 243-251.