International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-2, July 2019

# Artificial Neural Network (ANN) Based DDoS Attack Detection Model on Software Defined Networking (SDN)

**Pradeepa R, Pushpalatha M**

***Abstract:*** ***Software Defined Networking (SDN) is a modern emerging technology in networking. The great advantage of this network is, decoupling of the carrier plane and the control plane as well as which provides centralized control. A Controller is the intelligent part of SDN. It offers several benefits such as network programmability, dynamic computing, and cost-effective, high bandwidth. However, SDN has many security issues. The DDoS attack on SDN is a significant issue, and various proposals have been proposed for the detection and prevention of attacks. The main objective of this proposal is to detect DDoS attacks with the help of SDN techniques. In this proposal, a deep learning based Artificial Neural Network (ANN) model is used to detect the DDoS attacks. This can reduce learning time as well as detection time. To evaluate our model we use different machine learning algorithms and deep learning algorithm with different optimizers to train the network traffic which is generated in Mininet emulator and evaluates the results by various metrics such as detection rate, accuracy score, and confusion matrix with classification report. The result shows less detection time (4Secs) with a high accuracy score of 92% in our proposed Artificial Neural Network (ANN) model.***

***Keywords—Artificial Neural network, DDoS attacks, Machine Learning, Deep Learning, attack detection, Software-Defined Networks.***

## I. INTRODUCTION

Traditional networks are typically built with a huge number of devices like host machines, switches, routers and also packet forwarders, firewalls and so on with complex protocols. Network operators are responsible to manually change the control based on requirements.SDN is a perspective of programmable networks from recent developments [1]. The intelligence of an SDN network is logically controller and the network is divided into two parts such as control plane and the data plane. Control plane controls the entire network with its OpenFlow protocol [2] which contains flow information of the network data plan will forward packets as per the instruction of controller.As SDN beat the market of a network environment, security is the main agenda raised in SDN [3]. There are various types of attacks are there, classified by its vulnerability towards the device. Attacks are named by its implementation methodologies like DDoS, IDS.

Severe problems may arise when the DDoS with a group of attackers against the network devices. Many controllers are there in SDN based on applications supported by the controller is need more security.

The OpenDaylight controller has a lot of applications and features. It is an open-source java based controller [4]. We tested our network in Mininet emulator which contains switches, routers, and host devices and the results are same as the real network [5]. Machine learning algorithms are gaining more popularity in the field of network security to detect the attacks. More vulnerability is there in common machine learning models, need to do some better development is required for machine learning based SDN securities [6].

Many machine learning algorithms [7] are there to predict and detect network attacks based on the historical dataset, algorithms such as Support Vector Mechanism (SVM), K-neighbor classification (KNN), Logistic Regression (LR), Decision Tree (DT) and so on. Decision Tree (DT) [8] is a very popular and successful classification algorithm. Data can be described as a tree like structure for discrete or continues any kind of data. It performs a sequence of test with each internal node of the input attributes. Support Vector Mechanism (SVM) [8] algorithm based on the boundary between its two different types to provide enhanced classification. SVM can represent against complex functions. K-Nearest neighbor classifier (KNN) [9] with an optimal choice of k depends on its neighbors. Based on that k there will be a significant variation in different points. Logistic Regression (LR) [10] is a machine learning technique browed the field of statistics. It is used to learn the co-efficient logistic regression model of data and it is a sigmoid function.

The paper is designed in the following mannar, In the 2nd Section, related models are discussed. In section 3, we discuss the proposed model. 4th Section briefly discusses the performance study of our proposed model with the experimental setup, test scenario and result in analysis. Finally in the 5th section, concludes our proposed model and discussed some future works.

**Revised Manuscript Received on 30 July 2019.**
***** Correspondence Author
**Pradeepa R***, Department of Computer Science and Engineering, SRM Institute of Engineering and Technology, Chennai, India. pradimca@gmail.com
**Pushpalatha M**, Department of Computer Science and Engineering, SRM Institute of Engineering and Technology, Chennai, India . pushpalatha.m@ktr.srmuniv.ac.in

## II. RELATED WORKS

There are many proposals and methods are discussed about DDoS detection and prevention on SDN and SDN based environments.

DDoS attack threads cause degradation of network services resulting with vast loss in a network environment. Some of the proposed literatures related to our proposed model are reviewed in this section.

Methods, practices and solution to the DDoS attack detection and mitigation on SDN [11] in this proposal author clarify solutions and findings for detection and mitigation. They propose and present a proactive framework which is SDN based defense mechanism. They classified existing solutions according to its techniques and listed pros and cons of each model based on that classification they concluded there is some management rules and customizability is required for DDoS attack detection and prevention applications.

DDoS lightweight protection algorithm [12] is based on a set of rules to characterize data which is sent to the network as an attack or not. This lightweight algorithm mainly evaluates three criteria, such as CPU utilization, number of flow table entries and consumed bandwidth with POX controller. Time interval of the data collection process is also an important factor, if the interval is short then there will be an overhead on detection. In this proposal, they were more concentrated on CPU and bandwidth for lightweight scheme rather than the detection process. Also, they discuss block a botnet mounted DDoS attack [13], in this proposal they introduce efficiently block Mirai botnet mounted attacks.

Intelligent rule based DoS detection [14] model has two algorithms, one is feature selection algorithm and another is rule based classification algorithm. Scoring and ranking are used in the feature selection algorithm; classify the feature set based on the major or minor effect accordingly. Then the rule based classifications are used to detect the DoS attack based on priority selected by the feature selection algorithm. The list of rules is formulated by generic if-then rule. Achieved 98.5% of accuracy level in its detection algorithm but not discussed the classification time and detection time. More than detection accuracy classification and detection time is also important. Deep Learning for crossfire detection [15], in this proposal they discussed about different deep learning algorithms to detect and train the data. Controller capture traffic information and need to performance analysis, the higher frequent measurement will result in better detection rate. Based on that traffic data the deep learning algorithms are implemented to train the dataset. They compared ANN, CNN and LSTM networks algorithms and analysis the results. Almost all deep learning algorithms are approximately achieved 80% of accuracy with less detection time.

ASVM [16] Advanced support vector mechanism, objective of this proposal is to detect flooding based DDoS attacks on SDN. Volumetric and asymmetric techniques are used in this proposal to reduce the test and training time with best accuracy rate. In this proposal they organize customizable DDoS defense mechanism with alerts for security requirements. OpenDaylight multi controllers are used in their topology to generation traffic for training and validation data. Various metrics are utilized to analysis the performance of their proposed model.

This entire works discussed in this related literature survey; the existing systems are not satisfy high accuracy with low detection rate as well as light weight algorithm all together. As we conclude from this survey we need light weight rule based detection model with less detection time and high accuracy rate.

## III. PROPOSED MODEL

We created fat tree topology in Mininet environment and we generated normal as well as attack traffic and log the traffic dataset. Selected attributes from dataset is take as an input to our proposed model to calculate the detection output. Our proposed model is shown in fig. 1

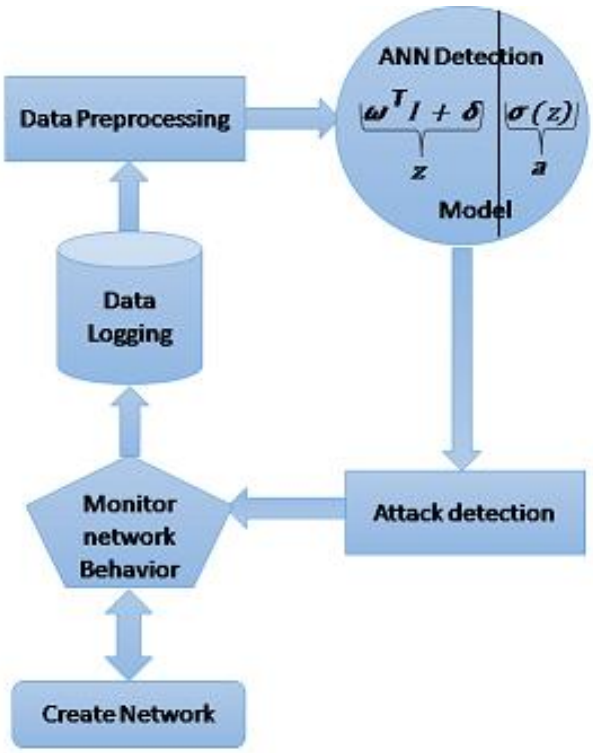



**Fig. 1 Proposed Model**

We propose an ANN model to detect DDoS attack on effective and efficient way with minimum detection time. Artificial neurons (node) are the collection unit of ANN. The weights of the edges help us to adjust the learning process. Connection between the artificial neurons is the edges, each connection transmit signal from one node to another node. Weight is calculated based on the signal strength of the each connection. The entire ANN is aggregated with layers.

### *A. Model computation:*

Proposed three layer neural network model is shown in Fig. 2. Input layer consist of 4 nodes, both hidden layers contain 5 nodes and the output layer consist of single node to generate single output.

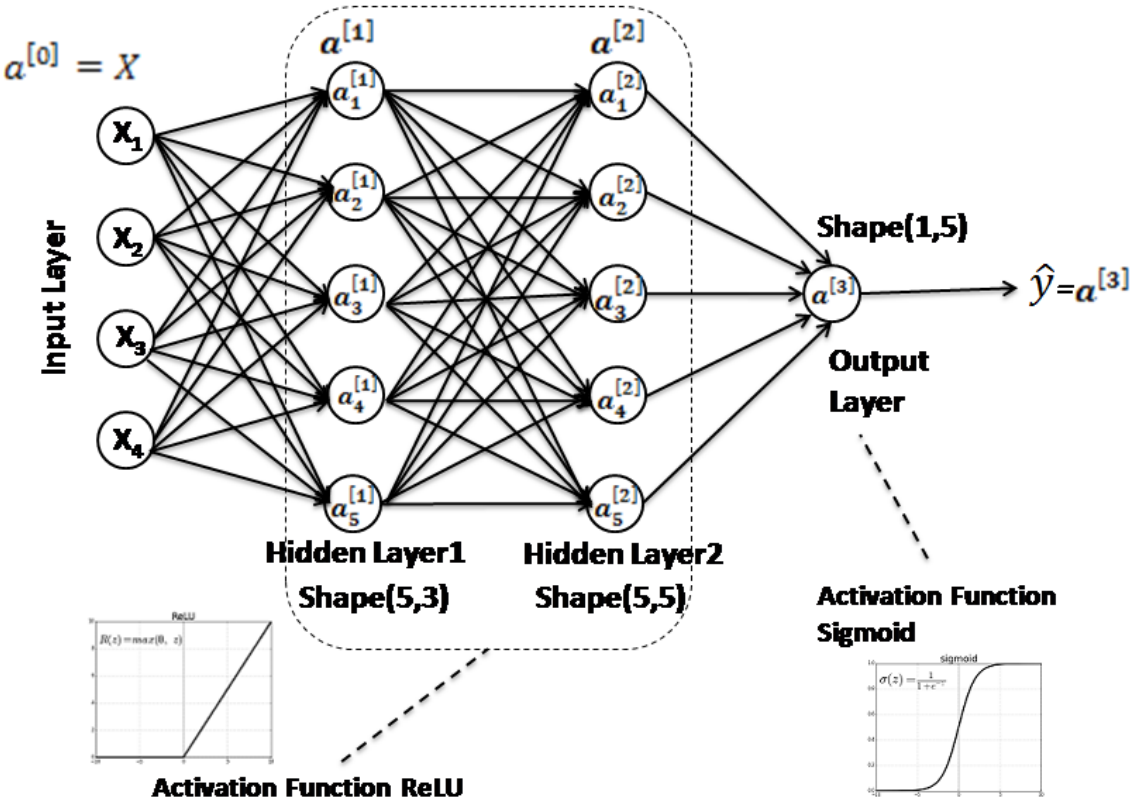



**Fig. 2 Proposed ANN Model**

Given inputs I, Where $I \in \mathbb{R}^{ni}$
Parameters: $\omega \in \mathbb{R}^{ni}$ *and* $\delta \in \mathbb{R}$
Output: $\hat{y} = \sigma(\omega^T i + \delta)$
Number of layers L in this proposed model is 3, units is each layer l is denoted by $n^{[l]}$ and the activation in each layer is denoted by $a^{[l]}$. Units and activation of the proposed model in each layer is, in Input layer 0: $n^{[0]} = n_x = 4$, hidden layers, layers 1 and 2: $n^{[1]} and\ n^{[2]} = 5$ and output layer 3 $n^{[3]} = 1$

***Input Layer:***

$$a^{[0]} = \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix} = I$$

***Hidden Layer 1:***

$$a^{[1]} = z^{[1]} = \omega_j^{[1]} a^{[0]} + \delta_j^{[1]}$$

The shape of the hidden layer 1 is formed with number of neurons in this layer and number of neurons in an input layer for our scenario shape of this layer is (5, 4), $\omega^{[1]}$ shape is (5, 5) with $\delta^{[1]}$ shape is (1, 1) and the matrix manipulation of hidden layer 1 is,

$$\begin{bmatrix} \omega_1^{[1]T} \\ \omega_2^{[1]T} \\ \omega_3^{[1]T} \\ \omega_4^{[1]T} \\ \omega_5^{[1]T} \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix} + \begin{bmatrix} \delta_1^{[1]} \\ \delta_2^{[1]} \\ \delta_3^{[1]} \\ \delta_4^{[1]} \\ \delta_5^{[1]} \end{bmatrix} = \begin{bmatrix} \omega_1^{[1]T} I + \delta_1^{[1]} \\ \omega_2^{[1]T} I + \delta_2^{[1]} \\ \omega_3^{[1]T} I + \delta_3^{[1]} \\ \omega_4^{[1]T} I + \delta_4^{[1]} \\ \omega_5^{[1]T} I + \delta_5^{[1]} \end{bmatrix} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \\ z_4^{[1]} \\ z_5^{[1]} \end{bmatrix}$$

***Hidden Layer 2:***

$$a^{[2]} = z^{[2]} = \omega_i^{[2]} a^{[1]} + \delta_i^{[2]}$$

$$a^{[2]} = z^{[2]} = \omega_i^{[2]} (\omega_i^{[1]} a^{[0]} + \delta_i^{[1]}) + \delta_i^{[2]}$$

The shape of the hidden layer 2 is formed with number of neurons in this layer and number of neurons in hidden layer 1 for our scenario shape of this layer is (5, 5), $\omega^{[2]}$ shape is (1, 5) with $\delta^{[2]}$ shape is (1, 1) and the matrix manipulation of hidden layer 2 is,

$$\begin{bmatrix} \omega_1^{[2]T} \\ \omega_2^{[2]T} \\ \omega_3^{[2]T} \\ \omega_4^{[2]T} \\ \omega_5^{[2]T} \end{bmatrix} \begin{bmatrix} \omega_1^{[1]T} I + \delta_1^{[1]} \\ \omega_2^{[1]T} I + \delta_2^{[1]} \\ \omega_3^{[1]T} I + \delta_3^{[1]} \\ \omega_4^{[1]T} I + \delta_4^{[1]} \\ \omega_5^{[1]T} I + \delta_5^{[1]} \end{bmatrix} + \begin{bmatrix} \delta_1^{[2]} \\ \delta_2^{[2]} \\ \delta_3^{[2]} \\ \delta_4^{[2]} \\ \delta_5^{[2]} \end{bmatrix} = \begin{bmatrix} z_1^{[2]} \\ z_2^{[2]} \\ z_3^{[2]} \\ z_4^{[2]} \\ z_5^{[2]} \end{bmatrix}$$

Activation function used in Layer 1 and 2 is ReLU

***Output Layer:***

$$a^{[3]} = \hat{y} = \sigma(z^{[2]})$$

***Forwarded propagation:***

Layer 0:

$I = a^{[0]}$

Layer 1

$$z^{[1]} = \omega^{[1]} i + \delta^{[1]}$$
$$a^{[1]} = g^{[1]}(z^{[1]})$$

Layer 2

$$z^{[2]} = \omega^{[2]} a^{[1]} + \delta^{[2]}$$
$$a^{[2]} = g^{[2]}(z^{[2]})$$

Layer 3

$$z^{[3]} = \omega^{[3]} a^{[2]} + \delta^{[3]}$$
$$a^{[3]} = \sigma^{[3]}(z^{[3]})$$

It can be generalized

$$z^{[l]} = \omega^{[l]} a^{[l-1]} + \delta^{[l]}$$
$$a^{[l]} = g^{[l]}(z^{[l]})$$

For $l = 1 \ldots 3$

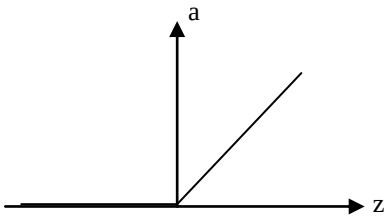
Where the shape of each parameter is calculated based on its units $n^{[l]}$

$\omega^{[1]} = (n^{[1]}, n^{[0]}) = (4, 5)$
$\omega^{[2]} = (n^{[2]}, n^{[1]}) = (5, 5)$
$\omega^{[3]} = (n^{[3]}, n^{[2]}) = (5, 1)$

***ReLU activation functions in hidden layers:***

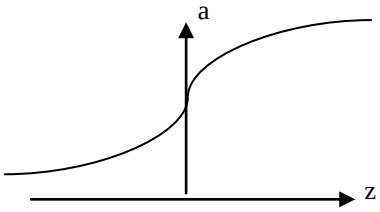
$g(z) = max(0,z)$

$$g^1(z) = \begin{cases} 0 \ if \ z < 0 \\ 1 \ if \ z \geq 0 \end{cases}$$




***Sigmoid activation functions in output layer:***

$$\sigma(z) = \frac{1}{1+e^z}$$




***B. Optimizers***

Various optimizers are tested with this model to improve the performance of the proposed model. Optimizers used are Adam, Nadam, sgd, and RMSProp.

**Adam**

Adam is an optimizer of the classical stochastic gradient descent algorithm to improve neural network link weights based on train data [17]. Adam merges the advantages of both Root Mean Square Propagation (RMSProp) optimizer and Adaptive Gradient Algorithm (AdaGrad) optimizer.

**SGD, NADAM and RMSProp**

Stochastic gradient descent (SGD) optimizer is supporting for momentum, learning rate and Nesterov momentum. Nesterov Adam optimizer (NADAM) is Adam RMSprop with Nesterov momentum. Root Mean Square Propagation (RMSProp) is normally a better choice for recurrent neural networks (RNN) [18].

## IV. PERFORMANCE ANALYSIS

***A. Experimental Setting***

The experiment of our proposed model is conducted on the OpenFlow enabled network based Mininet emulator and the topology is animated in MiniNAM. OpenDaylight controller is used in this network topology. The network topology used in this experiment is shown in Fig. 3. We generate 500 traffic data's in both normal and attack environments with 15 minutes of emulation time. We generated DDoS attack traffics and normal traffics are implemented in this work. Data collection is the most important task of this model to detect attacks on SDN. The network traffic data's are collected through OpenFlow switches. Collected data's are trained with machine learning and deep Learning algorithms, algorithms are developed and tested in python environment.

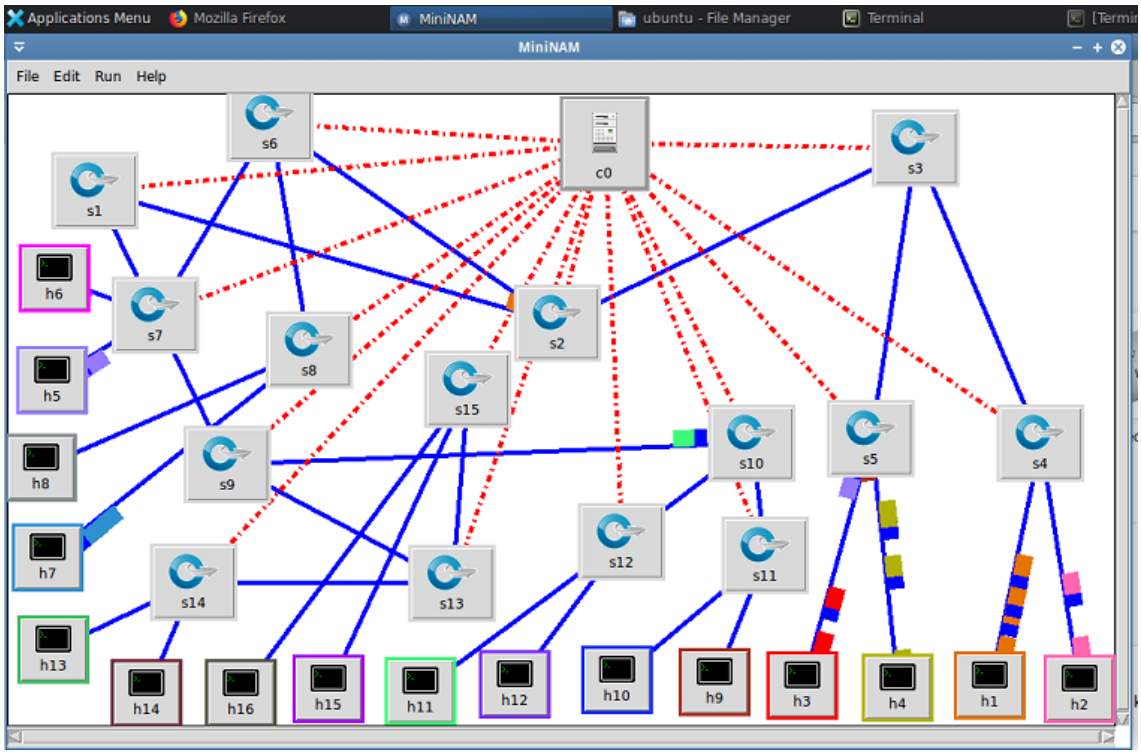



**Fig. 3 Network topology**

Our SDN topology contains 16 hosts, 15 switches, and 1 controller. The experiments are set up on OpenDaylight GUI Figure 4 shows our implemented topology.

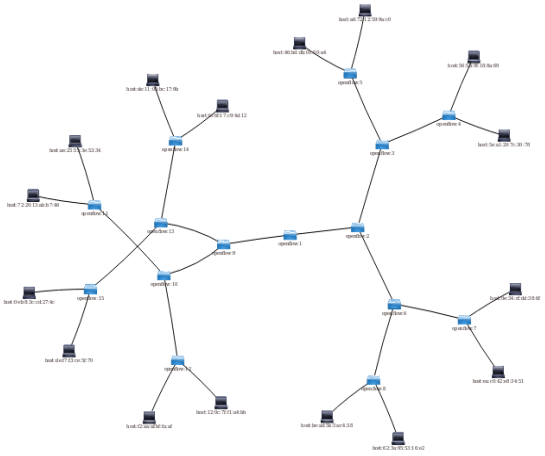


**Fig.4 Experimental setup in GUI**

### *B. Test Scenario*

Four different optimizers are verified to improve the performance of our proposed ANN model shown in table 1. Based on the following table Adam optimizer reduces the overall detection time and also training time in this proposed ANN.

**Table 1 Performance of Optimizers**

| Optimizer | Total Detection time (sec) | Training Time(ms)/ step | Loss |
|---|---|---|---|
| Nadam | 22 | 110 | 0.0809 |
| Adam | 4 | 20 | 0.0035 |
| Sgd | 28 | 141 | 1.603 |
| RMSProp | 18 | 92 | 0.6863 |

The classification among the optimizers are calculated based on accuracy, precision, recall and F1-score is shown in Fig.5 and based on the graph adam is a best optimizer which will help as to improve the performance of the proposed model.

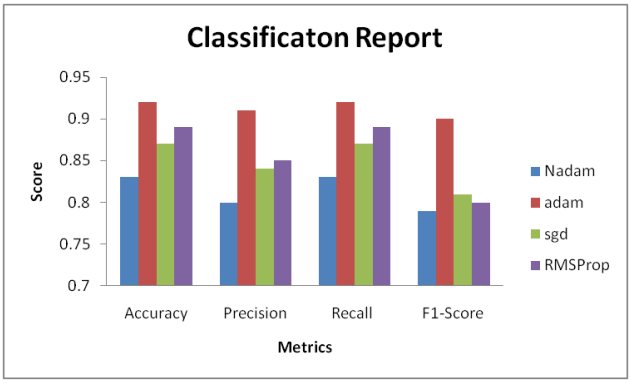



**Fig.5 Classification report among optimizers**

### *C. Evaluation Results*

The performance evaluation of proposed ANN model is compared with various machine learning models. Best model among the machine learning model is selected based on its cross validation report in Box plot shown in Fig. 6

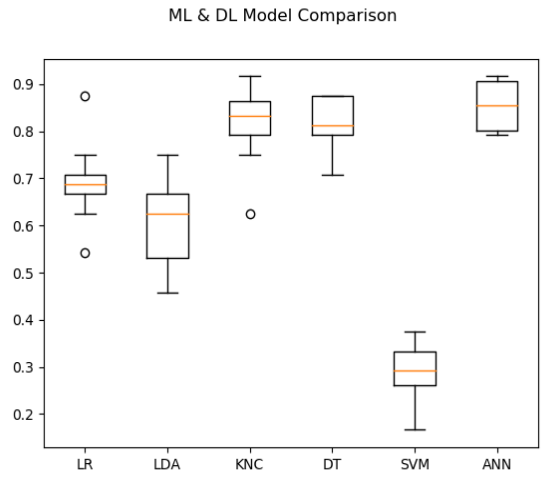



**Fig. 6 Model Comparison**

Based on the cross validation comparison Decision Tree (DT) and our proposed Model ANN is selected for next level analysis. Analysis was done with different metrics between ANN and DT.

### *Confusion Matrix*

It is a quick reference guide is used to illustrate the performance of classification algorithms. The following matrix table.2 describes the common confusion matrix format. Fig.7 and 8 shows the confusion matrix of ANN and DT accordingly.

**Table 2 Confusion Matrix**

| | Positive | Negative |
|---|---|---|
| TRUE | TP | TN |
| FALSE | FP | FN |

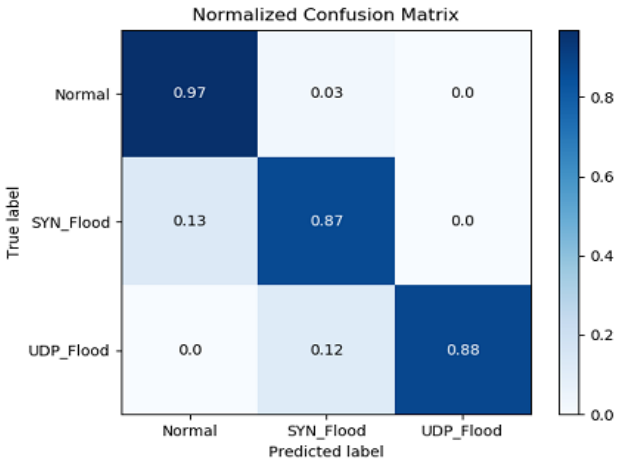



**Fig. 7 CNN Confusion matrix**

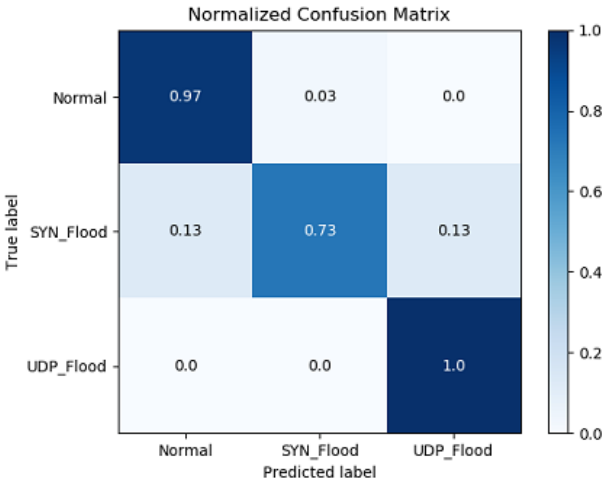



**Fig. 8 DT Confusion matrix**

### *Accuracy Score*

Accuracy is a great measure which calculated based on confusion matrix values, the accuracy score of ANN and DT is shown in table.3

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

**Table 3 Accuracy Score**

| Model | Accuracy Score (%) |
|---|---|
| DT | 88 |
| ANN | 92 |

### *Classification Report*

The performance of a model is evaluated based on classification metrics such as Accuracy, Precision, and Recall & F1 Score [19].

**Precision** is the ratio of correctly classified positive observations to the total classified positive observations. The precision graph between ANN and DT is shown in fig. 9

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

*Retrieval Number: B3670078219/19©BEIESP*
*DOI: 10.35940/ijrte.B3670.078219*
*Journal Website: www.ijrte.org*



*Published By:*
*Blue Eyes Intelligence Engineering*
*& Sciences Publication*

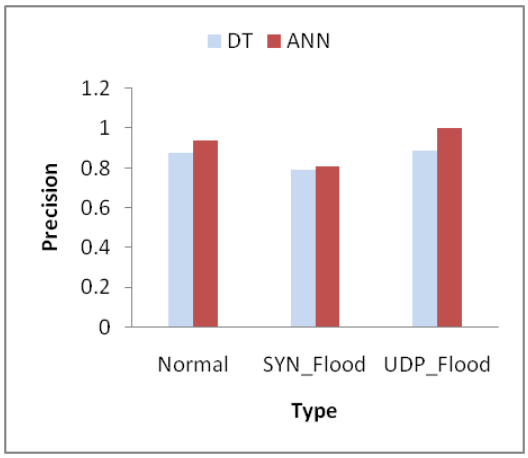



**Fig.9 Precision**

**Recall** is the ratio of correctly classified positive observations to the all observations in type. The Recall graph between ANN and DT is shown in fig. 10

$$Recall = \frac{TP}{TP + FN}$$

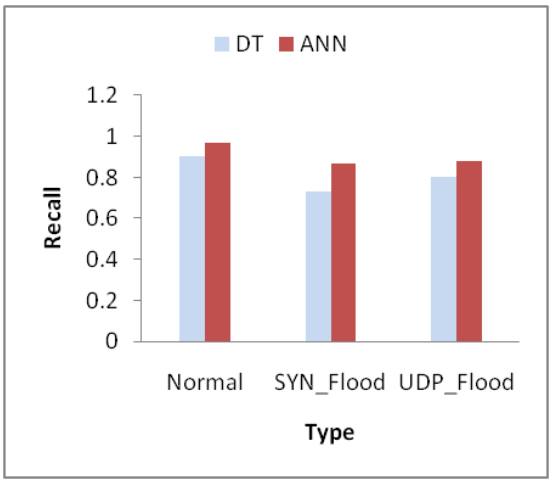



**Fig. 10 Recall**

**F1-Score** is the weighted average of precision and recall. The F1-Score graph between ANN and DT is shown in fig.11

$$F1 - Score = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

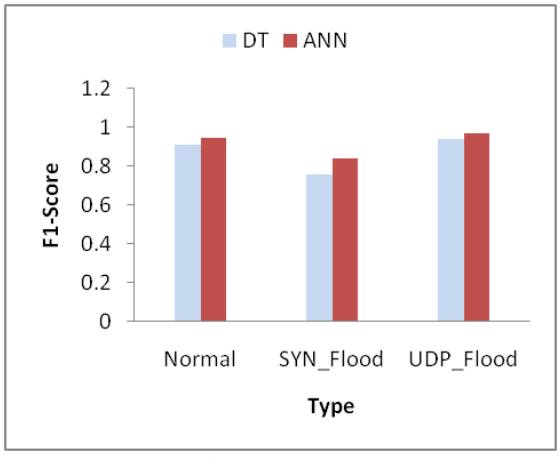



**Fig.11 F1-Score**

The accuracy score described approximately 92% of accuracy is achieved in our proposed model and we detect attack in 4sec and also less training time based on Adam optimizer in our proposed model. Classification report also shows the best performance of our proposed model.

## V. CONCLUSIONS AND FUTURE WORK

SDN is complementing the existing traditional networks by offering its features. Nevertheless, SDN Controller and Switch are vulnerable to DDoS attacks. In this proposal, we developed an ANN model to detect flood attacks of DDoS and we evaluated our proposed model with various machine learning algorithms with the help of three different metrics accuracy score, confusion matrix and classification report. Cross-validation method is also used to train and validate the method. The experimental results shows, the proposed model reaches high performance in terms of classification report with overall accuracy score of 92%. And also the detection time is reduced to 4Secs with the help of ADAM stochastic gradient descent optimizer. In future works we would like to comprise mitigation of DDoS attacks using best model.

## REFERENCES

1. Nunes, Bruno Astuto A., et al. "A survey of software-defined networking: Past, present, and future of programmable networks." IEEE Communications Surveys & Tutorials 16.3 (2014): 1617-1634.
2. https://noviflow.com/the-basics-of-sdn-and-the-openflow-network-architecture/
3. Scott-Hayward, Sandra, Gemma O'Callaghan, and Sakir Sezer. "SDN security: A survey." 2013 IEEE SDN For Future Networks and Services (SDN4FNS). IEEE, 2013.
4. S. Asadollahi, B. Goswami, and A. M. Gonsai, "Implementation of SDN using OpenDayLight controller," in Proceedings of the International Conference on Recent Trends in IT Innovations-Tec´afe, vol. 52, no .2, India, April 2017.
5. F. Keti and S. Askar, "Emulation of software defined networks using mininet in different simulation environments," in Proceedings of the 6th International Conference on Intelligent Systems, Modeling, and Simulation, Kuala Lumpur, February 2015.
6. Nguyen, Tam N. "The challenges in SDN/ML based network security: A survey." arXiv preprint arXiv:1804.03539 (2018).
7. Nanda, Saurav, et al. "Predicting network attack patterns in SDN using machine learning approach." 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). IEEE, 2016.
8. Latah, Majd, and Levent Toker. "Artificial intelligence enabled software-defined networking: a comprehensive overview." IET Networks 8.2 (2018): 79-99.
9. Balsubramani, Akshay, et al. "An adaptive nearest neighbor rule for classification." arXiv preprint arXiv:1905.12717 (2019).
10. Cui, Mingjian, Jianhui Wang, and Meng Yue. "Machine learning based anomaly detection for load forecasting under cyberattacks." IEEE Transactions on Smart Grid (2019).
11. Bawany, Narmeen Zakaria, Jawwad A. Shamsi, and Khaled Salah. "DDoS attack detection and mitigation using SDN: methods, practices, and solutions." Arabian Journal for Science and Engineering 42.2 (2017): 425-441.
12. Gkountis, Christos, et al. "Lightweight algorithm for protecting SDN controller against DDoS attacks." 2017 10th IFIP Wireless and Mobile Networking Conference (WMNC). IEEE, 2017.
13. Kolias, Constantinos, et al. "DDoS in the IoT: Mirai and other botnets." Computer 50.7 (2017): 80-84.
14. Rajendran, Rakesh, et al. "Detection of DoS attacks in cloud networks using intelligent rule based classification system." Cluster Computing: 1-12.

15. Narayanadoss, Akash Raj, et al. "Crossfire attack detection using deep learning in software defined its networks." 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring). IEEE, 2019.
16. Myint Oo, Myo, et al. "Advanced Support Vector Machine-(ASVM-) Based Detection for Distributed Denial of Service (DDoS) Attack on Software Defined Networking (SDN)." Journal of Computer Networks and Communications 2019.
17. https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/
18. https://keras.io/optimizers/
19. Sarang Narkhede "Understanding Confusion Matrix" May 9, 2018. https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62
20. https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/