Proceedings of 10th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2021 Month: October Volume: 10

Distributed Denial of Service Attack Detection on Software Defined Networking Using Deep Learning

Suwan Babu Bastola [a], Subarna Shakya [b], Susmita Sharma [c]

[a, b, c] *Department of Electronics and Computer Engineering, IOE, Pulchowk Campus, TU*
Corresponding Email:
[a] 074mscsk017.suwan@pcampus.edu.np, [b] drss@ioe.edu.np, [c] 074mscsk016.susmita@pcampus.edu.np

Abstract
The DDoS attack detection on Software Defined Networking architecture provides a central approach for monitoring the network traffics and informing the network administrator to apply respective counter measures. This research work builds different Deep Learning models for DDoS Detection viz LSTM, GRU, BLSTM and LSTM-GRU hybrid approach using the latest DDoS specific dataset CIC DDoS 2019. The comparison of the different detection models is done by cross-validation with the train-test split of 8:2. The hybrid LSTM-GRU model outperforms other models considering different performance metrics like accuracy, precision, recall, specificity and f-score. The LSTM-GRU and BLSTM detection model are implemented on the SDN architecture considering standard Carnet topology and different sized linear topology, and python based Ryu controller. The traffic including legitimate and DDoS traffic are generated on SDN environment is parsed in real time and values of the features is extracted, and fed into the detection model residing at SDN Ryu controller that classifies the traffic as normal or DDoS attack. The latency comparison shows LSTM-GRU model has lower latency than BLSTM model.On several SDN architectures, the LSTM-GRU based DDoS detection model is implemented. In terms of fault tolerance and CPU utilization %, the master-slave SDN design is proven to be more beneficial.

Keywords
Software Defined Networking, DDoS, CIC DDoS 2019, LSTM, GRU, BLSTM, TCP-SYN, UDP, Ryu Controller

1. Introduction

As more sensitive data becomes available on the Internet, security becomes a bigger issue in the networking industry. When data and information are sent from one system to another on the network, they must pass through a number of intermediary nodes, allowing other network users to access the data, putting the system's CIA aspect i.e. confidentiality, integrity, and availability at risk. In this assault, a huge number of infected workstations prevent genuine users from accessing web-based services. It is distinct from other denial of service (DoS) attacks in that it uses just one Internet-connected device (one network connection) to overwhelm a target with malicious traffic. At the network, transport, and application levels, DDoS attacks can be carried out via many protocols such as TCP, UDP, ICMP and HTTP.

1.1 Distributed Denial of Services(DDoS)

By flooding the target with fake traffic, DDoS is a malicious attempt to disrupt regular traffic to the online site, target server, service, or network. Because the host's computer resources are depleted, the target becomes unavailable to authorized users. DDoS attacks are mostly directed against the application layer, which is the layer of human-computer interaction where apps communicate with network services. The application layer is in charge of generating web pages on servers and responding to Hypertext Transfer Protocol (HTTP) requests.

1.2 Software Defined Network (SDN)

In traditional networks, control is distributed across all networking devices that make forwarding choices. Isolating control plane from those devices and centralizing in SDN, on the other hand, gives the following benefits. It makes it easier to decide

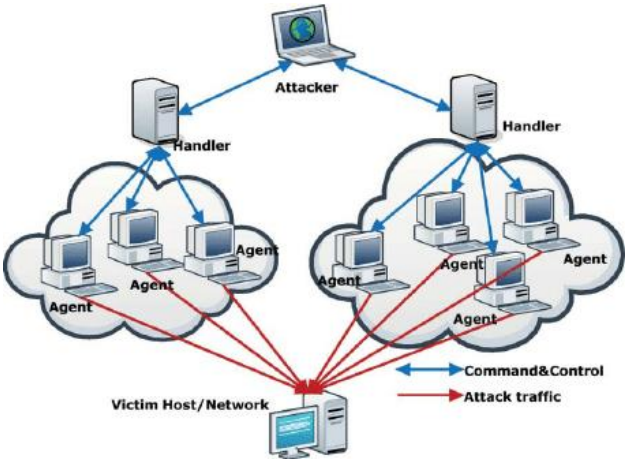

Figure 1: Flow Diagram of DDoS Attack

whether to accept or reject specific packets, assign priorities to them, and direct packet flow across the network, among other things. The control plane and data plane operate in a master-slave model, with the control plane acting as the master and the data plane acting as the slave. These networking devices act as forwarding devices by assessing the flow entries inserted and managed by the controller in flow tables. The Open Flow protocol is used to send messages between the controller and the data plane in a safe manner.

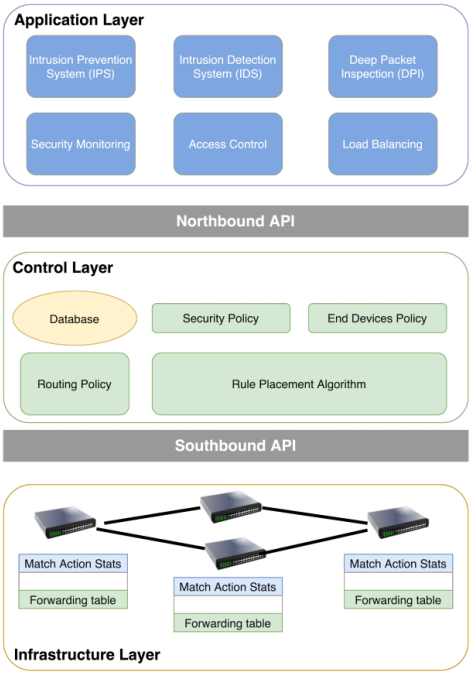

Figure 2: Software Defined Network

Northbound and Southbound interfaces are the two most important interfaces in the SDN network. A northbound interface connects external applications to the control plane. External applications can influence network behavior using this interface. The controller is connected to the physical data plane through the southbound interface. The SDN controller uses this interface to perform particular forwarding plane actions such as setting flow entries, rejecting packets, disconnecting the host from the network, and so on. Traditional networks have the same security flaws as SDN, as well as additional security problems. Because the controller in SDN is centralized, if an attacker gets control of it, severe problems arise. Firewalls, antivirus, and intrusion detection systems are some of the techniques created to filter out and identify different attacks, threats, and harmful activities.

2. Related Works

Different researchers have focused on distributed denial of service (DDoS) attacks, offering various detection strategies to fight DDoS flooding attacks. The many forms of DDoS attacks are due to the fact that DDoS flooding attacks have a wide range of features, making it difficult to identify with a single method[1]. When it comes to DDoS flooding attacks, a single source volume of data might be quite little, making it impossible to anticipate whether a request is good or malicious, resulting in detection systems with high positives or negatives rates[2].

Junhong Li discovers various dense neural network, DNN with autoencoder, and DNN with Pearson correlation coefficient models. When compared to standard neural networks, the model assesses performance using F1-score[3]. Within the TensorFlow Implementation framework, Peter Ken Bediako examined the performance of a LSTM deep learning for identifying DDoS flooding attacks where only volumetric attacks, such as TCP-SYNC, UDP, and ping attacks, were considered in the development of this model[4].

To understand the complicated connections among features, N. Shone et al. stacked two autoencoders. They argue that the soft-max layer is less effective than traditional classifiers, therefore they use a stacked auto-encoder in conjunction with a Random Forest classifier to identify intrusions[5]. Autoencoders were used by M. Al-Qatf et al. not only for feature learning but also to minimize the number of random variables considered. Instead of linking a

classifier to the autoencoder's output layer, a hidden layer representing compressed features is utilized as the classifier's input. The classifier is a Support Vector Machine (SVM). According to the authors, SVM surpasses all other traditional classifiers. Despite the fact that these offered approaches effectively solve the problem of feature selection, they do not address the difficulties of feature extraction[6].

Patil et al. suggested a Multithreaded Network Intrusion Detection System that is Protocol Specific. It is designed to detect DoS and DDoS attacks in cloud systems. It operates by using different classifiers dependent on the protocol of the incoming packet, such as the random forest algorithm, decision tree algorithm, and One R classifier. Experiments and conclusions reveal that the proposed design has a high degree of accuracy and a low percentage of false positives, but it fails to recognize a wide variety of attacks and offer real-time validation[7]. A. Saied et al. describe various systems around the Internet with infected zombies/agents that build botnets of networks in their article Detection of known and unknown DDoS attacks using Artificial Neural Networks. In this context, the objective of our effort was to recognize and neutralize existing and novel DDoS attacks in real-time scenarios. To identify DDoS attacks, they used an Artificial Neural Network (ANN) approach based on unique features that differentiate DDoS attack flow from genuine traffic[8].

In their article, "Protocol Specific Multi-Threaded Network Intrusion Detection System (PM-NIDS) for DoS/DDoS Attack Detection in Cloud," Rajendra Patil, Harsha Dudeja, SnehalGawade, et al. brought new dimensions to the realm of computer technology. Denial of service (DoS) and Distributed Denial of Service (DDoS) attacks have become serious threats to cloud technology in recent years. They propose a security architecture that is protocol specific Multithreaded Network Intrusion Detection System (PM-NIDS) for detecting DoS/DDoS attacks in the cloud in this article[9].

In their study "Detection of distributed denial of service using deep learning neural network," S. Sumathi and N. Karthikeyan explain why a neural network classifier is needed in an intrusion detection system for network security. For a publicly available dataset such as the KDD Cup, DARPA 1999, DARPA 2000, and CONFICKER datasets, this study assesses network performance using a deep learning neural network classifier with a cost reduction technique. The performance study is based on performance parameters such as detection accuracy, cost per sample, average latency, packet loss, overhead, packet delivery ratio, and throughput. In comparison to existing algorithms, the simulation results show that the DNN Cost minimization algorithm provides better results in terms of high detection accuracy 99 percent with less false reduction, high average delay, less packet loss, less overhead, high packet delivery ratio, and high throughput [10].

P. S. et al. created a new technique for intrusion detection to classify the NSL-KDD dataset by combining a genetic algorithm (GA) for optimum feature selection and LSTM in their paper, using a LSTM-RNN to classify Network Attacks. According to the study, the LSTM-RNN classifiers with the best feature set improve intrusion detection. The IDS's performance was assessed using parameters like the accuracy, recall, precision, F score, confusion matrix. The classifiers' performance was evaluated using the NSL-KDD dataset. Classifying NSL-KDD datasets were into binary as normal and abnormal, and multi-class sets using an LSTM-RNN. This experiment concludes the LSTM-RNN-with-GA model outperformed the random forest by 10[11]. Tuan et al. analyzed the CIC DDoS 2017 dataset for features such as source and destination IP addresses, source and destination ports, flow duration per second, packet duration per second, bytes per second, and packet duration and concluded that the GRU-RNN model outperforms SVM and DNN modes for detecting DDoS attacks[12].

This research focuses on DDoS attacks and proposes machine learning models with feature selection techniques for detecting attacks, as well as mitigation strategies. As a result, the goal is to create DDoS attack detection systems and use deep learning to apply the model in an SDN-based architecture.

3. Research methodology

The design of the SDN topology, as well as the development of detection models utilizing various machine learning approaches, are all part of the development of the DDoS attack detection system in Software Defined Networking. Finally, comparing the performance of the models created in the Software Defined Networking Framework.

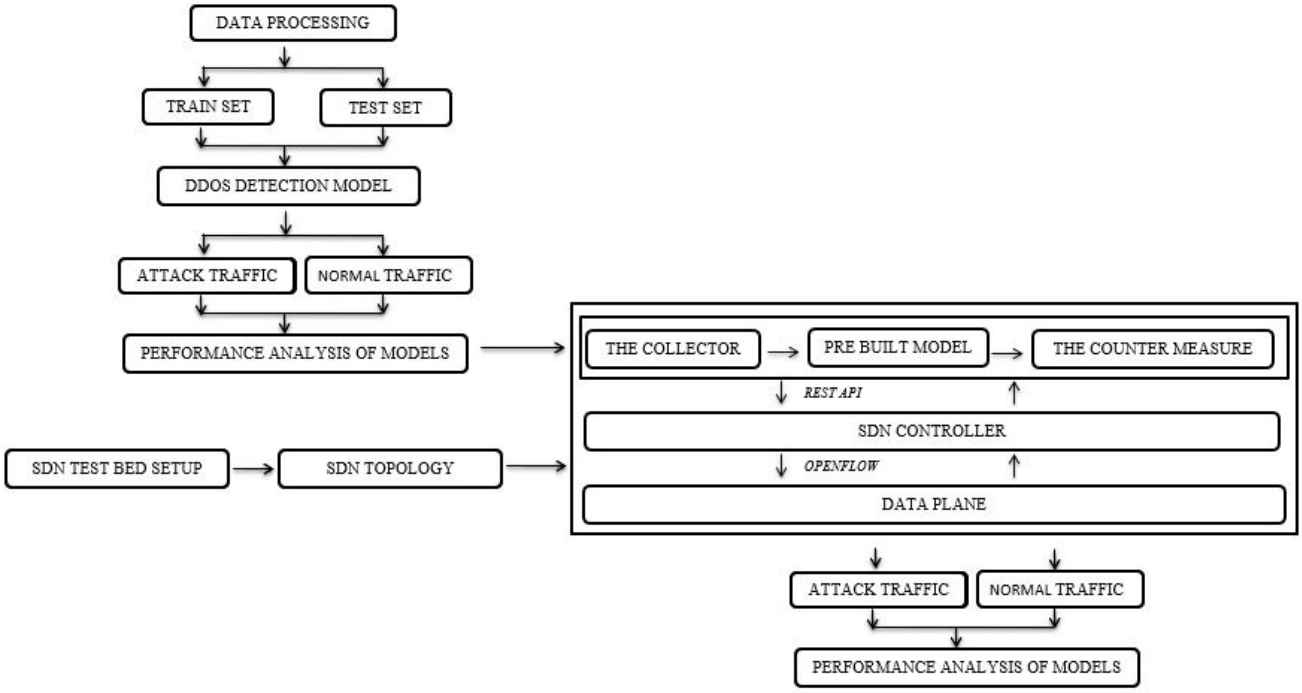

Figure 3: The System Block Diagram

3.1 DDoS Attack Detection Model

Recurrent Neural Networks are a form of Neural Network in which the output from the previous step is used as input in the current phase. In classic neural networks, all inputs and outputs are independent of one another, however in some situations, it is more important to forecast the next word of a phrase than it is to remember the prior words, thus remembering the past words is necessary. As a result, RNN was created, which can address this problem with the aid of a Hidden Layer. RNN remembers certain information about a sequence using Hidden state, which is the network's main characteristic. Short-term memory issues can be solved using the LSTM and GRU approaches. They feature internal devices known as gates that control the flow of data. These gates contain features that determine whether or not data in a sequence should be kept or discarded. The machine learning models are trained by the train dataset. The various RNN models like GRU, LSTM, BLSTM and LSTM-GRU hybrid models are trained.

Gated Recurrent Unit: The Recurrent Neural Network model consists of reset gate and current memory gate along with an update gate that determines whether or not to send the previous output to the following cell.

$$z_t = \sigma^*(W_Z[h_{t-1}, x_t]) \tag{1}$$

$$r_t = \sigma^*(W_r[h_{t-1}, x_t]) \tag{2}$$

$$h_t^{\sim} = \tanh^*(W[r_t * h_{t-1}, x_t]) \tag{3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * h_t^{\sim} \tag{4}$$

Where the input x_t, previous cells output $h_t - 1$, value for the next cell$h_t$and weight W.

Long Short Term Memory: The model consists of four different gates:Forget Gate(f), Input Gate(i), Input Modulation Gate(g) and Output Gate(o).

$$f_t = \sigma * (W_f[h_(t-1), x_t] + bf) \tag{5}$$

$$i_t = \sigma * (W_i[h_(t-1), x_t] + bi) \tag{6}$$

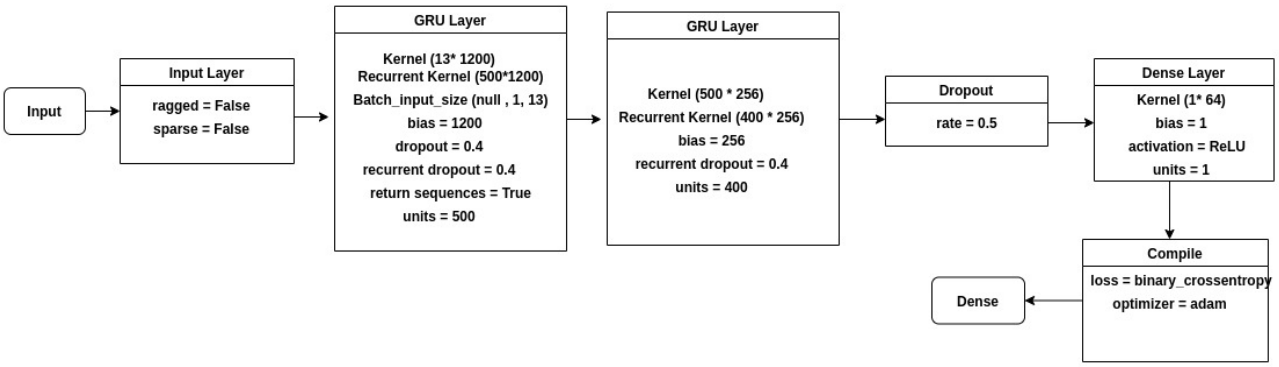

Figure 4: The Implementation of GRU Model

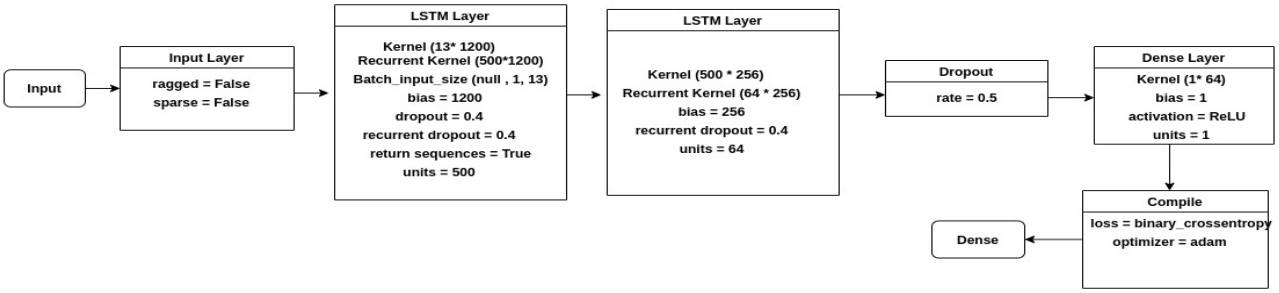

Figure 5: The Implementation of LSTM Model

$$C_t^{\sim} = \tanh * (W_c[h_{t-1}, x_t]) + b_c \tag{7}$$

$$C_t = f_t * C_{t-1} + i_t * C_t^{\sim} \tag{8}$$

$$o_t = \sigma * (W_o[h_{t-1}, x_t] + b_o) \tag{9}$$

$$h_t = o_t * \tanh(C_t) \tag{10}$$

Where x_t is the input, (C_{t-1}) , $(h_{t-1})$is the output from the previous cell, h_{t-1} is the value for the following cell, and W is the weight.

Bidirectional LSTM: Bidirectional LSTMs are a kind of LSTM that enhances model performance in sequence classification tasks. When all timesteps of the input sequence are accessible, BLSTMs train two LSTMs on the input sequence instead of just one. The first is based on the original input sequence, while the second is based on a reversed duplicate of the original input sequence. As a consequence, it will be able to give more context to the network, resulting in faster and more complete learning on the topic.

At each time step t, BLSTM computes the forward LSTM layer output and the backward LSTM layer output separately, and then concatenates these values to obtain the BLSTM output. The updating equations of BLSTM are:

$$h_t^{\longrightarrow} = LSTM(x_t, h_{t-1}^{\longrightarrow}) \tag{11}$$

$$h_t^{\longleftarrow} = LSTM(x_t, h_{t-1}^{\longleftarrow}) \tag{12}$$

$$y_t = W_y^{\longleftarrow} * h_t^{\longleftarrow} + W_y^{\longrightarrow} * h_t^{\longrightarrow} + b_y \tag{13}$$

Hybrid LSTM GRU Model: The hybrid LSTM GRU model for the detection of the DDoS attack includes the LSTM followed by the GRU model. The hybrid model including LSTM and GRU has better performance that the individual models. So, it will be implemented for DDoS Detection Model development.

3.2 DDoS Detection Model in SDN Framework

The Ryu controller resides on the control layer, which will extract the essential features from new packet's

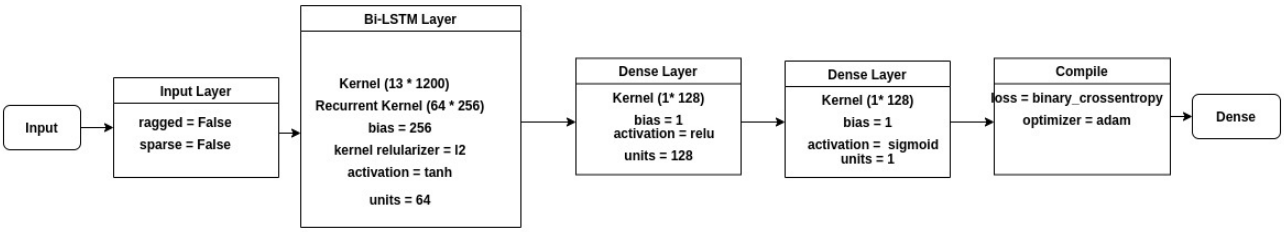

Figure 6: The Implementation of BLSTM Model

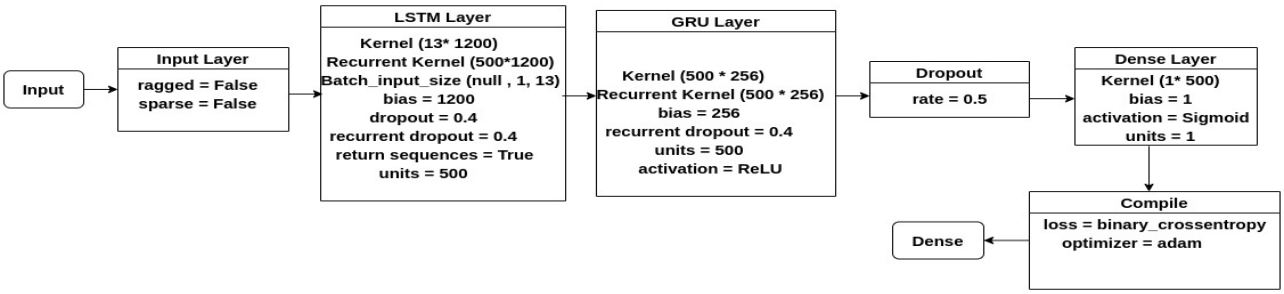

Figure 7: The Implementation of LSTM-GRU Model

Packet In message and forwards to the DDoS detection model. The DDoS detection program will determine whether or not it is an assault. It will inform the controller of the relevant choice. If the traffic is being dropped because of an attack, the controller will create a suitable flow table for sending that packet to the Open Flow Switch's on-flow entries

SDN Test Bed Setup: The SDN network experiment testbed is built on an Intel computer with virtual machines for Mininet and Ryu controllers. The network is controlled by the Ryu controller, which is an SDN controller. On Mininet, there are eighteen hosts linked to the OpenFlow virtual switches. Mininet provides the platform for virtual test bed and allows development environment for SDN. It is a network simulator that creates network with hosts, controller, switches and links between them. The controller is assigned with IP 192.168.0.101, likewise each host are assigned with the IP address like H1 with network of 10.0.0.1, mac address of 00:00:00:00:00:01 and in similar way to other hosts.

3.3 Implementation of DDoS Detection Model on SDN

Infrastructure Layer This infrastructure layer includes the set of Open Flow Switches which are connected to the different devices. They receive the traffic from the devices and, decides the flow on the basic of flow entries of the Open Flow Switches. The

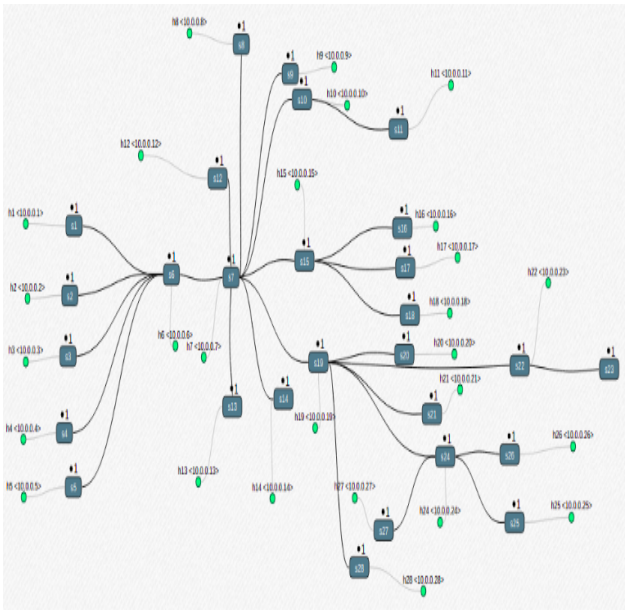

Figure 8: The Implementation of Standard Carnet Topology in SDN Framework

normal traffic generation includes basic communication kinds of TCP and UDP traffics. The TCP, UDP, and ICMP traffic are generated using the iperf and ping commands. The commands used are listed below:-
TCP-SYN Traffic
iperf -p port address -c
ip address of http server UDP Traffic
iperf -p port address -u -c
ip address of http server

The DDoS attack program Hping3 is also used for the creation of DDoS attacks. Hping3 can receive and send data packets by specifying the binary or string format of the data packets in the TCL language. UDP Flooding hping3 -2 -V –rand-source
–flood random destination . TCP-SYN Flooding hping3 -S -V -p 80 –rand-source
–flood ip address of HTTPServer.
The collection of the attributes is done at Ryu-controller. The packet it receives during different normal traffic and attack traffics, it parses and get values for the different attributes of the dataset. The Hping3 tools will be used to evaluate the DDoS attack detection system on the SDN framework. Basic communication types such as TCP and UDP traffic are included in regular traffic production. Hping3 is a DDoS attack program that generates unusual network traffic. Hping3 can receive and send data packets by specifying the binary or string format of the data packets using the TCL language.

SDN Controller The OpenFlow protocol is used to manage an OpenFlow switches from the remote Ryu controller which is Southbound Interface. Using different messages such as packet out, alter flow table, and so on, this protocol controller may create, update, and remove flow entries on OFS. Packet in, flow removed messages are used by OFS to interact with the controller.

Application Layer The DDoS attack detection model thus formed will resides at the application layer of the SDN framework. It has the collector, predefined detection module and the counter measure for the traffic accordingly. From the data received from the controller, the collector gathers the values of many variables such as Flow ID, Source IP address, Source Port address, Destination IP address, Destination Port, Protocol detail, Timestamp, Flow Duration, Flow Bytes/second, and Flow Packets/second. These properties values are saved in the file. The values of these attributes are taken from a file, scaled and reshaped, and then fed into a pre-built DDoS Detection model, which determines if the traffic is an attack or not. The countermeasure device will inform the controller if the traffic is normal or if it is under assault.

3.4 The Dataset

CIC DDoS Dataset The CIC DDoS 2019 dataset is the most recent dataset, and it includes several forms of DDoS attacks as well as normal traffic. This dataset's exploitation-based attacks are being investigated in the creation and detection of an SDN-based DDoS model. TCP-SYN, UDP, and UDP latency attacks are all included in this attack. In TCP SYN Flooding Attack, the attacker exploits a vulnerability in the TCP connection sequence, specifically three-way handshaking, in a TCP-SYN flood. The SYN request is made during three-way handshaking to start a TCP connection with the host. A SYN-ACK answer is required in response to this SYN request. Then the requester confirms it with an ACK answer. A SYN flood happens when a requester sends a high number of SYN requests but doesn't react to the host's SYN-ACK response or sends the SYN inquiries from a bogus IP address. At such cases the host system waits forever for each request to be acknowledged, causing resource constraints until no new connections can be created and, finally, denial of service. In UDP Flooding Attack, UDP flood occurs when an attacker floods a target system with UDP packets. This attack's objective is to overwhelm a remote host with random ports. As a result, the host will periodically check for the application listening on that port. The victim server's ports will be exhausted as a result of this operation, resulting in a denial of service. UDP-lag Flooding Attack is a form of attack that breaks the connection between the server and the client. The opponent player slows or interrupts the movement of competing players in this sort of assault, which is mostly employed in gaming. It can be done out by software that operates on a network system and consumes the bandwidth of the other participants. After the dataset processing i.e. concatenation, feature extraction, cleaning, transformation and train-test split the dataset consists of 8827912 rows. It is split into 7062330 training entries and 1765582 for testing entries with 14 attributes.

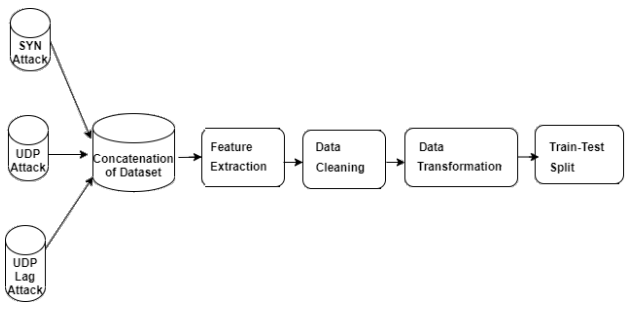

Figure 9: The Dataset Pre Processing

4. Result

DDoS attacks are detected using a variety of machine learning algorithms. These model's performance is compared using parameters such as loss and accuracy. Training is done for 50 epochs during the model's training phase. Each model's loss and accuracy variations are investigated.

The accuracy and loss graph for the training and testing of different machine learning model shows that the hybrid model LSTM-GRU has the highest accuracy and minimum loss per epoch considering 50 epochs. Also the analysis of different models was done where hybrid model shows the better result in terms of accuracy, recall, specificity, precision and F-score as shown in table 1.

Table 1: Comparison of Neural Network Models.

S.N.	Model	Accuracy	Recall	Specificity	Precision	F-Score
1	GRU	79.00	64.995	79.873	16.768	26.659
2	LSTM	94.026	55.965	96.408	49.240	52.388
3	Bi-LSTM	95.956	76.840	97.149	62.707	69.057
4	LSTM-GRU	98.579	80.615	99.700	94.383	86.957

4.1 Testing of Normal Traffic

The normal traffic is generated from the host connected at the SDN OpenFlow switches by different commands.

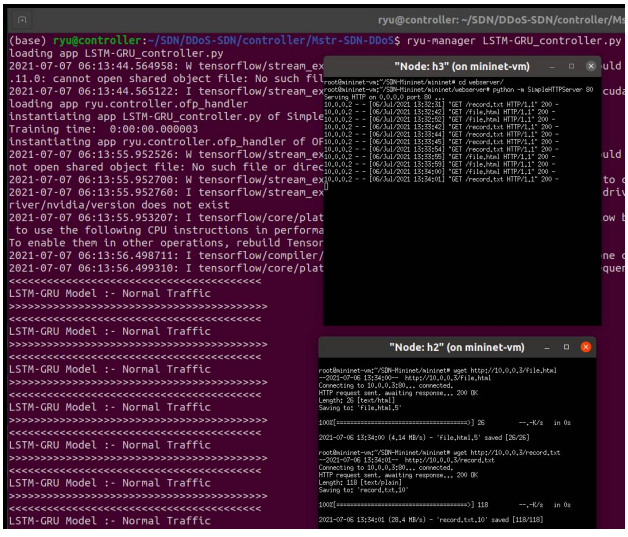

Figure 10: Detection of Normal Traffic

4.2 Testing of DDoS Traffic

The attack traffic is generated from the host connected at the SDN OpenFlow switches by different commands The UDP attack are generated by using

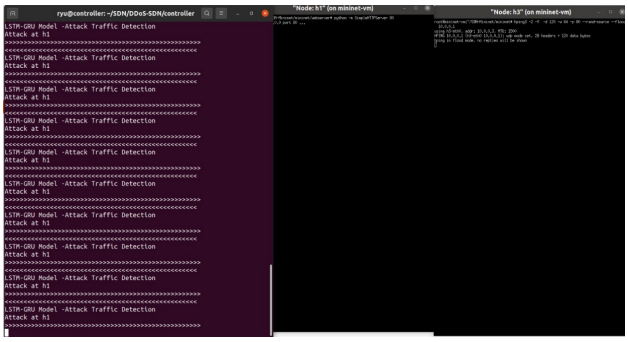

Figure 11: Detection of UDP Attack

hping3 commands from different host. The detection model shows DDoS attack traffic. As the UDP attacks,

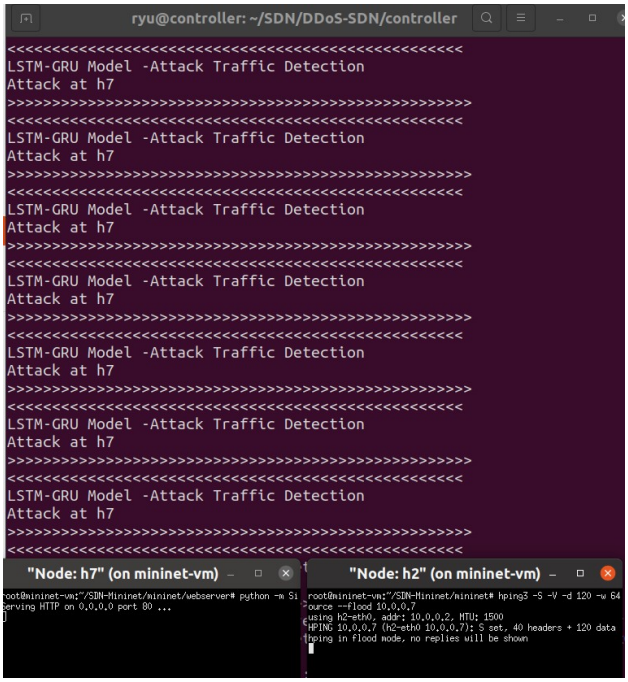

Figure 12: Detection of TCP Attack

TCP-SYN DDoS attacks are generated from the different hosts using hping3 commands, and the detection model shows attack detected.

4.3 Comparison of Model's performance on different topologies

The different linear SDN topologies with different number of switches and host are considered to analyze the performance of different models on the basis of latency parameter.

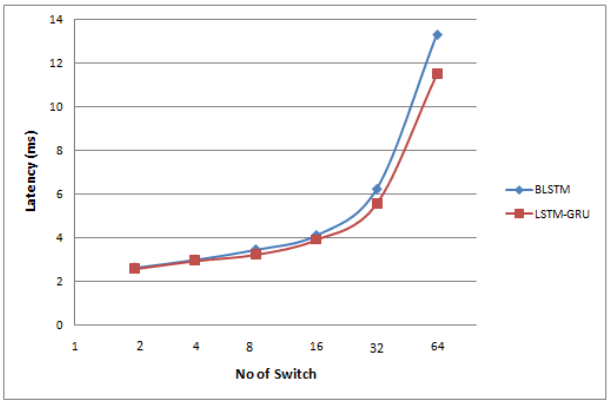

Figure 13: Latency variation with increasing number of switches

The Carnet topology from topology zoo consisting of 27 OpenFlow switches each consisting a host is also separately implemented and analysed. Again the simulation result shows that the LSTM-GRU model has lower latency than the BLSTM model.

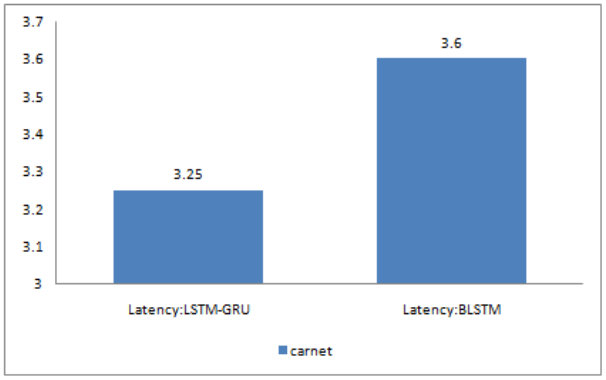

Figure 14: Latency of models on Carnet topology

4.4 Single and Multi-Controller SDN Framework

We evaluate the performance of the selected LSTM-GRU model on the standard Carnet topology on three SDN scenarios: single controller, master-slave controller and dual-equal controller. CPU utilization is taken as the comparison parameter which is calculated using the ubuntu 'top' command. In case of single SDN controller we put only one controller. We generate flood of DDoS traffic and evaluate the CPU utilization of the RYU controller. The CPU utilization of The single controller is 20% and this architecture is not fault tolerant.

In equal role two SDN controller architecture, we install two controllers that separately monitors the network but they do not know each other. Both the

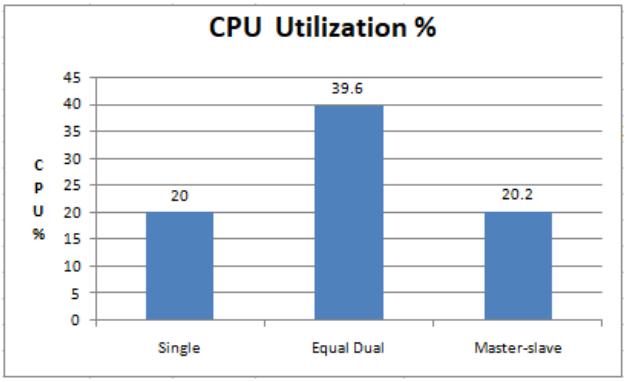

Figure 15: CPU utilization % of LSTM-GRU Controller

controllers connects to all nodes separately and does the controlling task independently. This makes the system fault tolerant but with high resource utilization. In the master-slave architecture of the controller, the master controller actively monitors the whole network where as the slave controller remain silent. These two controllers communicates with each other via east-west protocol. Here a Echo server is maintained by the master controller through which the communication between controllers become possible. During the failure of master controller, the slave controller automatically performs the role of master controller. As soon as the master controller is available, again the controller responsibility is taken by it.

The figure 15 shows the comparative result of CPU utilization % of LSTM-GRU based DDoS detection controllers in three different scenarios. Result show that equal-role two controller architecture has the highest CPU utilization. The single controller has the smallest CPU utilization. The master-slave model has the intermediate CPU utilization.

Conclusions

The performance of different models based on F-score, accuracy, recall and precision the result shows that BLSTM and LSTM-GRU has better results. The detection model is tested on the SDN Architecture which is created using a python-based Mininet-Ryu controller. The legitimate and DDoS traffic is generated by several hosts using Hping3 and its flooding option. The controller classifies the traffic as either normal or DDoS attack. The SDN based DDoS Detection model performed the real time detection of attacks. Simulation data shows that in the

case of small number of switch there is no significant change in the latency but along with increment of the number of switches in the network, the latency also increases. The increase of latency of BLSTM model is higher than that of LSTM-GRU model. On several SDN architectures, the LSTM-GRU based DDoS detection model is implemented. In terms of fault tolerance and CPU utilization %, the master-slave SDN design is proven to be more beneficial.

References

[1] Monowar H Bhuyan, Hirak Jyoti Kashyap, Dhruba Kumar Bhattacharyya, and Jugal K Kalita. Detecting distributed denial of service attacks. *The Computer Journal*, 57(4):537–556, 2014.

[2] ABM Alim Al Islam and Tishna Sabrina. Detection of various denial of service and distributed denial of service attacks using rnn ensemble. In *2009 12th International Conference on Computers and Information Technology*, pages 603–608. IEEE, 2009.

[3] Junhong Li. Detection of ddos attacks based on dense neural networks, autoencoders and pearson correlation coefficient. 2020.

[4] Peter Ken Bediako. Long short-term memory recurrent neural network for detecting ddos flooding attacks within tensorflow implementation framework. Master's thesis, Luleå University of Technology, Computer Science, 2017.

[5] Quamar Niyaz, Weiqing Sun, and Ahmad Y Javaid. A deep learning based ddos detection system in software-defined networking (sdn). *arXiv preprint arXiv:1611.07400*, 2016.

[6] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. *Eai Endorsed Transactions on Security and Safety*, 3(9):e2, 2016.

[7] Jitendra Kumar, Rimsha Goomer, and Ashutosh Kumar Singh. Long short term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters. *Procedia Computer Science*, 125:676–682, 2018.

[8] Majjed Al-Qatf, Yu Lasheng, Mohammed Al-Habib, and Kamal Al-Sabahi. Deep learning approach combining sparse autoencoder with svm for network intrusion detection. *IEEE Access*, 6:52843–52856, 2018.

[9] Alan Saied, Richard E Overill, and Tomasz Radzik. Detection of known and unknown ddos attacks using artificial neural networks. *Neurocomputing*, 172:385–393, 2016.

[10] Rajendra Patil, Harsha Dudeja, Snehal Gawade, and Chirag Modi. Protocol specific multi-threaded network intrusion detection system (pm-nids) for dos/ddos attack detection in cloud. In *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–7. IEEE, 2018.

[11] S Sumathi and N Karthikeyan. Detection of distributed denial of service using deep learning neural network. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–11, 2020.

[12] Tuan A Tang, Lotfi Mhamdi, Des McLernon, Syed Ali Raza Zaidi, and Mounir Ghogho. Deep learning approach for network intrusion detection in software defined networking. In *2016 international conference on wireless networks and mobile communications (WINCOM)*, pages 258–263. IEEE, 2016.