Hybrid Deep Learning Based DDoS Detection System Using Software Defined Networking

Arvind T[1], Prof.K.Radhika[2]

[1] Department of CSE, OU, Hyderabad, Telangana ,India
[2] Professor, Department of AI & DS, CBIT, Hyderabad, Telangana, India

ABSTRACT

The Software-Defined Networking (SDN) paradigm exhibits substantial potential, as it enables the centralized controller to dynamically configure and manage networks. The SDN paradigm's programmable and open characteristics have led to the emergence of novel security challenges, such as Distributed Denial of Service (DDoS) attacks. The impact of DDoS attacks can impede network services, diminish network efficiency, and result in significant harm to the overall network infrastructure in the realm of SDN. The utilization of machine learning and deep learning algorithms has become a prevalent method for detecting DDoS attacks, owing to their efficacy in analyzing voluminous data sets and detecting patterns or anomalies that signify the presence of an ongoing attack. The aim of this study is to create a dataset that is customized for SDN by incorporating the acknowledged features into a CSV file. This study introduces a novel technique for detecting DDoS attacks: the Hybrid Deep Learning Model-Based DDoS Detection System (HDL3DS). This model employs Autoencoder and Deep Neural Network techniques to detect DoS attacks with reliability. HDL3DS's performance was assessed and compared to that of other DL models, such as MLP, LSTM, and DNN. In accordance with the experimental findings, the proposed HDL3DS outperformed other models, obtaining a 99.98% accuracy rate.

Keyword: *SDN, DDoS, Deep Learning, Machine Learning, Ryu, Mininet.*

1. INTRODUCTION

SDN is a promising network architecture that enables dynamic and programmable network administration via a centralized controller. However, the open and flexible nature of SDN also introduces new security challenges, such as DDoS attacks, which can severely disrupt network services and degrade network performance. DDoS attacks in SDN can take various forms, including volumetric attacks that overwhelm the network with traffic, protocol attacks that exploit vulnerabilities in network protocols, and application-layer attacks that target particular applications and services. The identification and prevention of DDoS attacks in SDN is essential for network service availability and reliability. In conventional networks, machine learning models and deep learning models have demonstrated encouraging outcomes in identifying DDoS attacks, and their implementation in SDN environments is on the rise. Large quantities of network data, such as flow statistics, packet identifiers, and controller logs, can be analyzed by machine learning algorithms to automatically identify patterns and anomalies that indicate DDoS attacks. SDN-based DDoS detection systems can adapt to changing attack patterns and provide real-time detection and mitigation by leveraging the power of machine learning. Using machine learning and deep learning models for purpose of detecting DDoS attacks in SDN has several advantages in this context. First, these algorithms can detect DDoS attacks in real-time, allowing for prompt mitigation and response. Second, they are capable of adapting to changing attack patterns and updating their detection capabilities accordingly. Thirdly, they can offer valuable insights into the characteristics and patterns of DDoS attacks, thereby facilitating the creation of more effective countermeasures. Integration of machine learning and deep learning with other SDN security mechanisms, such as access control and authentication, can provide a comprehensive defense against DDoS attacks.

2. RELATED WORK

The dataset was generated by the authors [5], and it possesses distinct characteristics. It consists of 65,000 instances of DDoS and 64,000 instances of regular traffic, each containing 12 feature characteristics and a single labeled class. The hping3 tool was utilized to produce DDoS traffic. The machine learning models employed in this study for training and testing the datasets included SVM, NB , KNN), and ANN, in conjunction with feature selection techniques. According to the evaluation results, the KNN model utilizing wrapper feature selection achieved the highest level of accuracy at 98.3%. The author has put forth a proposal for an intrusion detection system (IDS) that utilizes machine learning (ML) techniques to identify instances of TCP-SYN flooding attacks.

Nam et al. [23] employed a statistical approach combined with neural network technology to discover abnormal network activity. The researchers employed the entropy measure to select attributes from a given list and subsequently utilized SOM to characterize the behavior of the network. The experiment was done using a POX controller and a Mininet emulator. The accuracy and computational overhead of the model were examined.

Matheus PNovaes [11] et al. developed the LSTM-FUZZY model for identifying and mitigating DDoS and port scan attacks in SDN. Three steps comprise the system’s operation: characterization, anomaly detection, and mitigation. The approach was assessed in two different scenarios: first on IP traffic collected from floodlight controllers and mininets, and then subsequently on the CICD DoS 2019 dataset. The proposed system was evaluated in comparison to SVM, KNN, MLP, LSTM-2, and PSO-DS. The results of the evaluation indicate that the proposed model surpassed the other by reaching an accuracy of 96.22 percent.

Isa and Mhamdi[20] et.al proposed a hybrid unsupervised machine learning technique utilizing auto-encoding for the purpose of detecting intrusions in Software-Defined Networks (SDNs).Measurements of throughput and latency are used to evaluate the controller’s efficacy in combination with the deployed model. The results obtained from the Nsl-KDD dataset indicate that the SDN controller's performance is minimally impacted, while achieving a detection accuracy of 98.4.

As a means of detecting DDoS attacks in SDN, a deep convolution neural network (CNN) ensemble system was proposed by Shahzeb Haider[22] et al. Four characteristics (B. packet length, avg packet size, flow duration, and flow inter_arval_time) were taken from the CICIDS-2017 dataset and used to evaluate the proposed model. There are 140,000 observations in total, with normal and attack traffic making up 60% and 40%, respectively, of the total. With the help of the RELU and sigmoid activation functions, four DL-based architectures (RNN, LSTM, CNN, and ensemble CNN) were built. To conduct the experiments, they used the Keras package, which has a TensorFlow backend. A number of metrics, such as accuracy, precision, recall, F1, training time, testing time, and CPU use, were used to judge the proposed models. The proposed model was accurate to the tune of 99.45%. Similar approach was adopted by authors [21] to detect slow DDoS attacks using hybrid CNN-LSTN. The method's performance is assessed using customized datasets. The experimental findings revealed that the suggested model outperformed all other models on all considered metrics.

Researchers Phan [2] et al. utilized the Distributed Self-Organizing Map (DSOM) technique within OpenFlow switches as a means of addressing the bottleneck issues in the controller of extensive networks. The findings of this research indicate that the employment of a DSOM results in a higher level of detection accuracy with significantly reduced overhead as opposed to the utilization of a singular SOM.

The authors [7] presented a deep learning methodology utilising the Gated Recurrent Unit (GRU) to address the issue of DDoS attacks in SDN environments. The classification of DDoS attacks has been conducted using a limited set of six features that were selected from the NSL-KDD dataset. The authors asserted that their proposed model attained an accuracy of 89%.

Dehkordi [19] et.al proposed a hybrid approach that employs statistical and machine learning methodologies to address DDoS attacks. The study employed an analytical approach for feature extraction and a machine learning technique for classification. The authors validate their methodology by employing the UNB-ISCX, CTU-13, and ISOT datasets.

Deepa and colleagues [17] introduced an ensemble approach for identifying abnormal network traffic patterns within the SDN controller. The ensemble was enhanced for improved efficiency through the utilization of KNN, Naive Bayes, SVM, and self-organizing maps. The authors validate their approach through the utilization of Mininet. The researchers utilized a combination of KNN-SOM, NV-SOM, and SVM-SOM algorithms in their study. Their findings indicate that the SVM-SOM algorithm outperforms the others in terms of detection rate and accuracy.

3. IMPLEMENTATION ENVIRONMENT AND METHODOLOGY

The study involved conducting experiments on a virtual machine that was based on Linux and operated on 64-bit Ubuntu 20.04 LTS. The virtual machine was equipped with a Core-i5 processor and 8 GB of RAM and was installed on a VMware workstation. The experimental setup includes the utilization of Mininet [16], an open-source software for creating virtual networks, along with Ryu [14-15], an OpenFlow SDN controller implemented in Python. Additionally, the implementation involves the use of Python, Keras, Tensorflow, and Jupyter Notebook. We use the dataset [1,4,12] with additional statistical features such as, no_conn, bi_flows, unq_ip_per_dst, the dataset contains 25 features. The table displays the traffic distribution of the Dataset.

Traffic	ICMP	UDP	TCP	Total
Normal	206101	121200	177714	505015
Attack	152353	143795	165547	461695
Total	358454	264995	343261	966710

Table-1 Dataset Traffic Distribution

Datpath_id	Switch identification
Timestamp	Packet arrival time
Flow_id	Flow identification
Src_ip	IP addr of source
Dst_ip	IP addr of destination
Tp_src	Source port no
Tp_dst	Destination port no
Ip_proto	Protocol type
Icmp_code	ICMP code
Icmp_typ	ICMP type
Flow_dur_sec	Flow duration in seconds
Flow_dur_nsec	Flow duration in nano sec
Hard_timeout	Flow table entry expiration time regardless of packet matching

Idle_timeout	Flow timeout provided no packet match
Flags	connection status
In_port	Input port
Byte_count	No.of bytes
Byte_count_per_sec	No.of bytes per second
Byte_count_per_nsec	No.of bytes per nano seconds
Packet_count	No.of packets
Packet_count_per_sec	No.of packets per second
Packet_count_per_nsec	No.of packets per nano seconds
no_conn	No .of connections
unq_ip_per_dst	no.of unique source ip's per dst ip
bi_flows	Bi-directional flow
Label	0 for legitimate , 1 for Malicious

Table-2: Dataset-feature-description

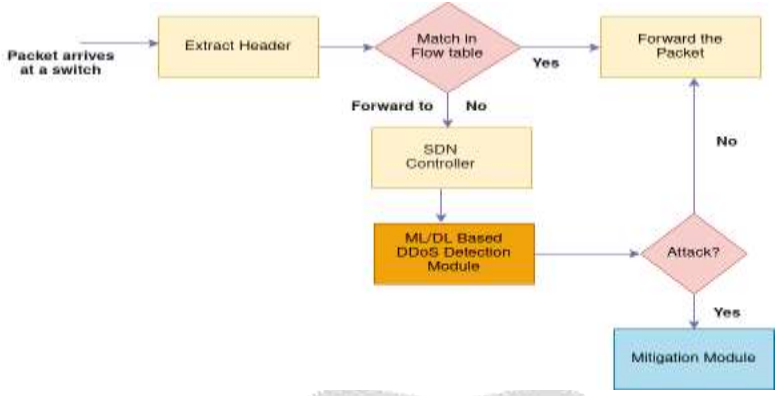

Fig-1: Detection-Mitigation-Flow

The fig.1 shows the flow of detection and mitigating process.

4. DEEP LEARNING MODELS

4.1 Multilayer Perceptron Layer Algorithm

The ANN approach known as the Multilayer Perceptron (MLP) is widely used for supervised learning tasks like classification and regression. The MLP is composed of one or multiple hidden layers of neurons, where each neuron establishes connections with all neurons in the preceding and succeeding layers. A weighted sum of inputs is calculated, an activation function is applied, and the weights and biases are updated depending on the difference between the expected and actual output via forward and backward propagation. The process of training a neural network involves iteratively performing forward and backward propagation, while the prediction task entails feeding input features through the network that has undergone training.

For each layer of forward propagation, the following equations must be solved:

z_i = w_i * x + b_i (1)

a_i = activation_function(z_i) (2)

Where:

- z_i represents the inputs for the i^{th} neuron in the layer as a weighted sum
- w_i denotes the weights that are connected to i^{th} neuron in the layer
- x denotes the input features
- b_i is the bias term associated with i^{th} neuron in the layer
- activation_function(z_i), is utilised to calculate the activation value a_i of a neuron by applying it to the weighted sum of inputs.

4.2 LSTM Algorithm

The Long Short-Term Memory (LSTM) is a recurrent neural network architecture that is often employed in time series data processing, particularly in the detection of network anomalies. LSTM models can learn to detect abnormal patterns in network traffic by analyzing the temporal patterns in the data.

4.3 Auto Encoder Algorithm

Autoencoders are a class of unsupervised learning algorithms that, given some input data, can encode that data into a lower-dimensional space and subsequently decode it to recover the original data. The primary aim of an autoencoder is to reduce the discrepancy between input and output by minimizing the reconstruction error.By training an autoencoder on normal network traffic, it can detect anomalies such as DDoS attacks by comparing the reconstruction error of incoming traffic with the learned representation.

4.4 Deep Neural Network Algorithm

Deep neural networks (DNNs) have the potential to be employed in the identification of network anomalies. This can be achieved by training the DNN on the data pertaining to normal network traffic and subsequently utilizing it to classify incoming traffic as either normal or anomalous. The DNN is commonly comprised of multiple layers of densely interconnected neurons. These neurons receive input features and execute a series of transformations on them, ultimately generating output predictions.
A DNN is formally expressed as a function that accepts an n-dimensional input vector x and creates an m-dimensional output vector y. The function can be represented as a sequence of embedded nonlinear transformations in the format:

$$y = f(Wn(f(Wn\text{-}1(...f(W_1x + b_1)...)+b_{n-1}))+b_n) \qquad (3)$$

Where W1,...,Wn are weight matrices, b1,...,bn are bias vectors, and f is a nonlinear activation function, which can be either the sigmoid or ReLU function. The final layer, which takes the output of each preceding layer as its input, generates the resultant vector y.
To train a DNN, its weights and biases are optimized to reduce the cost function representing the difference between the expected and actual output. Stochastic gradient descent and other optimization methods are often utilized for this purpose.

4.4 Proposed Algorithm

This work proposes the Hybrid Deep Learning Based DDoS Detection System (HDL3DS) to detect DDoS attacks. This is based on autoencoder and DNN algorithms. After preprocessing the input data it is fed into autoencoder model.

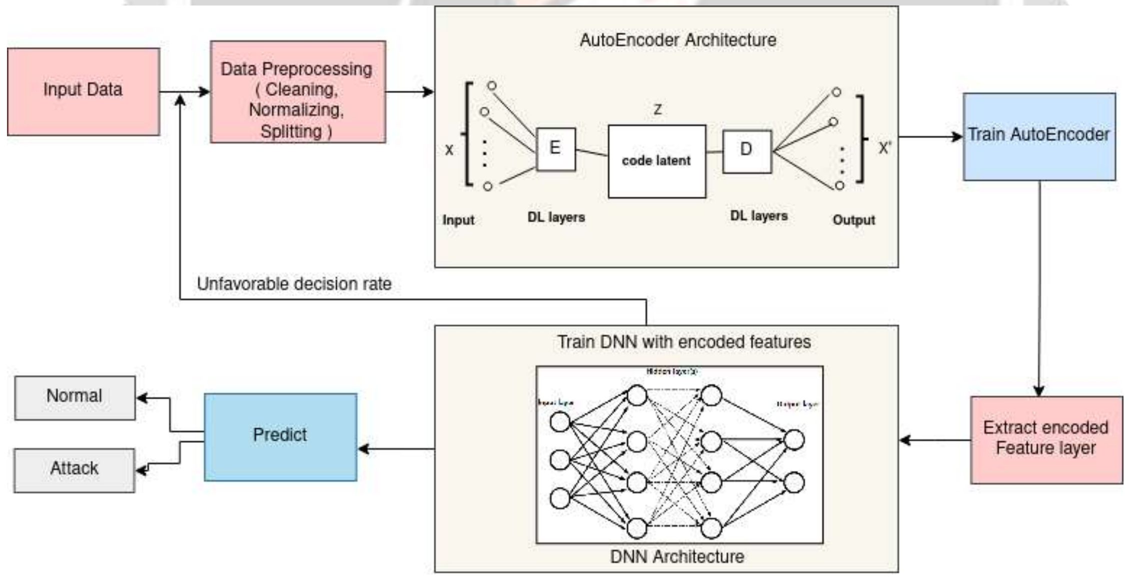
Input Data
Data Preprocessing (Cleaning, Normalizing, Splitting)
AutoEncoder Architecture
x
E
Z
code latent
D
X'
Input
DL layers
DL layers
Output
Train AutoEncoder
Unfavorable decision rate
Train DNN with encoded features
DNN Architecture
Extract encoded Feature layer
Predict
Normal
Attack

Fig-2: HDL3DS architecture

Representation of the HDL3DS model:

Let X be the input data of shape (n, m), where n denotes the total no. of observations and m represents the no. of attributes.

Autoencoder Encoder:

- The encoder takes the input ‘X’ and produces a latent representation ‘Z’ that has a shape of (n, d), where ‘d’ denotes the size of the latent space.
- The encoder function can be represented as

  f_enc(X) = Z, (4)

  Where f_enc is the encoder function.

Autoencoder Decoder:

- The decoder function utilizes the latent representation ‘Z’ as input and generates a reconstructed version of the original input ‘X’, which has dimensions (n, m).
- The decoder function can be represented as

  f_dec(Z) = X', (5)

  where f_dec is the decoder function and X' is the reconstructed input.

Autoencoder Loss:

- The loss function of the autoencoder model is represented as MSE(X, X'), where MSE stands for mean squared error. This function calculates the difference between the original input X and the reconstructed input X'.
- The autoencoder loss can be represented as

  L_AE(X, X') = (||X – X'||^2), (6)

  Where ||.||^2 denotes the squared Euclidean distance.

DNN Classifier:

- The DNN utilizes the latent representation ‘ Z ‘ to generate a prediction for a given classification task.
- The DNN function can be expressed as

  f_DNN(Z) = Y, (7)

  Where ‘f_DNN ’ denotes the DNN function and ‘ Y ’ represents the predicted output.

DNN Loss:

- Joint optimization of autoencoder and DNN losses is the primary training goal.
- The joint training objective can be mathematically expressed as

  L_total(X, Y_true) = L_AE(X, X') + alpha * L_DNN (Y_true, Y), (9)

- The DNN loss can be represented as

  L_DNN (Y_true, Y) = (-1/n * sum (Y_true * log(Y) + (1 - Y_true) * log (1 – Y))). (8)

Training Objective:

- Joint optimization of autoencoder and DNN losses is the primary training goal.
- The joint training objective can be mathematically expressed as

  L_total(X, Y_true) = L_AE(X, X') + alpha * L_DNN (Y_true, Y), (9)

  Where alpha is a hyperparameter that controls the weight of the DNN loss in the overall loss function.

The parameters of the autoencoder and DNN are jointly optimized to minimize the aggregate loss L_total.

Parameter	**Optimal value**
Input size	25
Encoding size	8
Hidden layers	3
Encode Activation function	Relu
Decode Activation function	Sigmoid
Number of Epochs	30
Loss function	binary_cross_entropy
Optimization type	Adam
Batch size	64, 128

Table-2: Autoencoder parameters

Parameter	**Optimal value**
Hidden layers	3
Number of channels (neurons)	8
Activation function	Relu,sigmoid
Number of Epochs	30
Loss function	binary_cross_entropy
Optimization type	Adam
Batch size	64, 128

Table-3: HDL3DS parameters

5. EXPERIMENTAL RESULS

The tabular representation illustrates a comparison of the models' performance metrics.

Model	Accuracy	Precision	Recall	F1
MLP	61.35	61.09	100.0	75.85
LSTM	59.0	52.0	59.0	50.0
DNN	90.39	91.05	93.34	92.18
HDL3DS	99.98	99.99	99.85	99.92

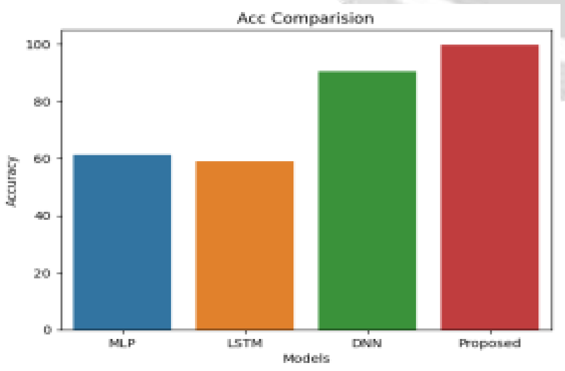

Fig-3: Acc-comparison of the models

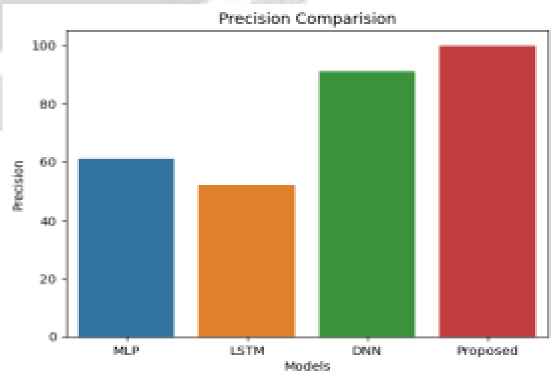

Fig-4: Precision-comparison of the models

The fig-3 and table.4 illustrates a comparison of the accuracy of various classifiers. It is evident that the proposed Hybrid detection model outperforms the DNN, MLP, and LSTM in terms of accuracy. Furthermore, the model under consideration attained a accuracy rate of 999.98%. Despite producing an accuracy rate of 59%, the LSTM model fails to produce satisfactory results. This can be attributed to the fact that LSTMs are more complex than MLP and DNN models. Models with an increased number of parameters require more computational resources. Due to the limited availability of training data in DDoS datasets, complex models such as LSTMs may be susceptible to over fitting. The DNN model was 90.39% accurate, while the MLP model was 61.35% accurate.

The fig-4 and table.4 depicts a precision comparison among various classifiers. It is evident that the proposed hybrid detection model exhibits higher precision in comparison to DNN, MLP, and LSTM. Furthermore, the model under consideration attained a precision rate of 99.99%. Although the LSTM model yields 52.0% of accuracy, the DNN model exhibits a precision of 91.05%, while the MLP model demonstrates a precision of 61.09%.

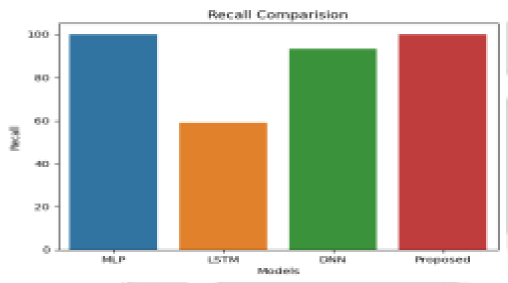

Fig-5: Recall-comparison of the models

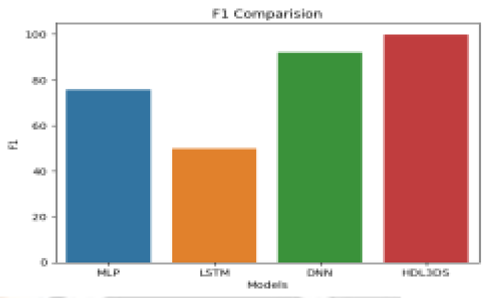

Fig-6: F1-comparison of the models

The fig-5 depicts the recall values obtained by different classifiers in the current study. In terms of recall, it is evident that the proposed hybrid detection model and MLP models outperforms the DNN and LSTM. In addition, the recall rate of MLP was 100%, while the proposed model achieved a recall rate of 99.85%. This result can be attributed to the data preprocessing techniques that were utilized. The model's success can be attributed to the synergistic combination of the individual components' strengths and the benefits of a collaborative learning strategy. Although the LSTM model produced a recall value that was 40% lower than that of the proposed model, the DNN model demonstrated a recall rate of 93.34%.

Fig- 6 displays the F1-Measure values attained by the classifiers utilised in this investigation in comparison to other models. In the current investigation, it was observed that the F1-Measure of the LSTM model only attained a value of 50%. Conversely, the proposed model achieved a significantly higher F1-Measure of 99.92%.

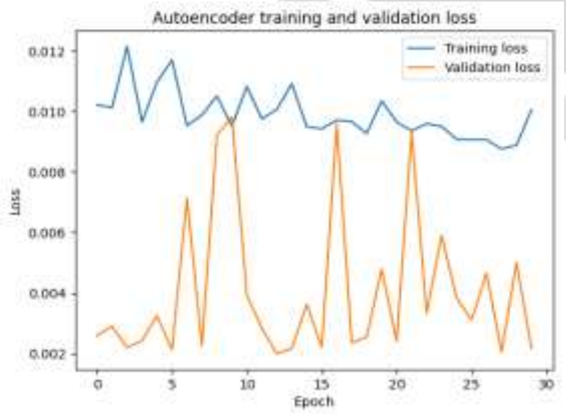

Fig-7: Autoencoder training and validation loss

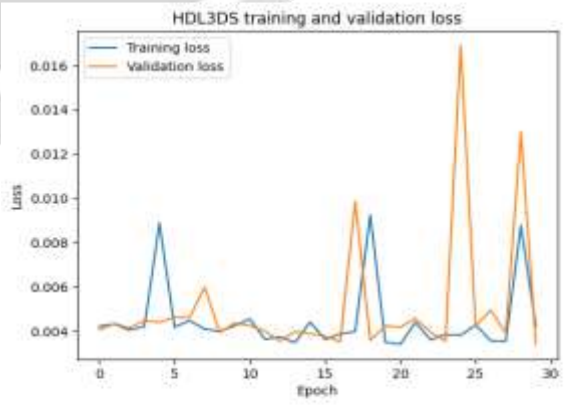

Fig-8: HDL3DS training and validation loss

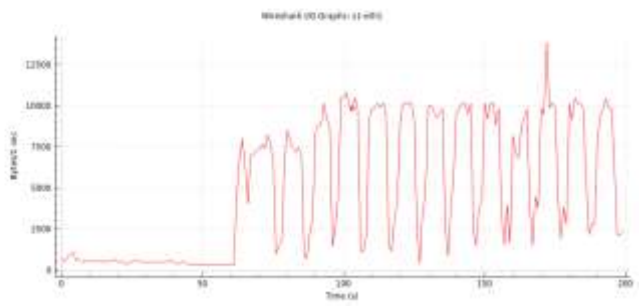

Fig-9: Normal-attack-byte-rate

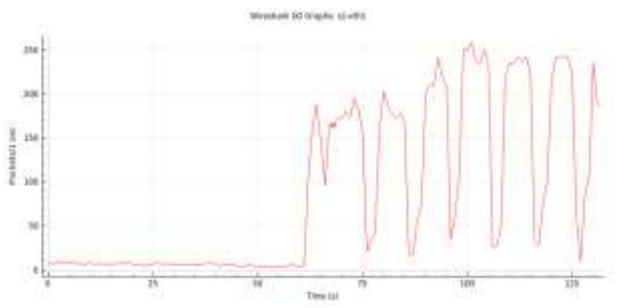

Fig-10: Normal-attack-packet-rate

The fig-7 dipicts the Training loss and validation loss of auto encoder model, these are two frequently utilized metrics for monitoring the performance and convergence of a model. The training loss, which is also referred to as reconstruction loss or reconstruction error, quantifies the degree of dissimilarity between the initial input data and the reconstructed output generated by the autoencoder throughout the training process. The primary objective of the autoencoder is to minimize the loss function, thereby acquiring the ability to reconstruct the input data with the utmost accuracy. The validation loss metric serves as an approximation of the model's ability to generalize, denoting its efficacy in reconstructing data that has not been previously encountered or is considered out-of-sample. The model's training loss and validation loss were recorded as 0.010 and 0.002, respectively, over the course of 30 epochs. While Fig.8 depicts the training and validation losses of HDL3DS, which were observed to be 0.005 and 0.003 over 30 epochs, respectively,

The fig-9 depicts the byte rate in the context of both normal and attack traffic. It is evident that the byte rate for normal traffic ranged between 500 and 700 bytes per second, whereas for attack traffic, it was observed to be in the range of 10000 to 12500 bytes per second.

The fig-10 depicts the packet rate in the context of both normal and attack traffic. It is evident that the byte rate for normal traffic ranged between 10 and 50 packets per second, whereas for attack traffic, it was observed to be in the range of 50 to 250 packets per second.

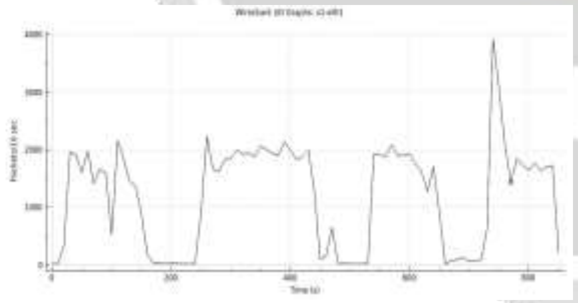

Fig-11: Packet-rate-attk-counter-measure

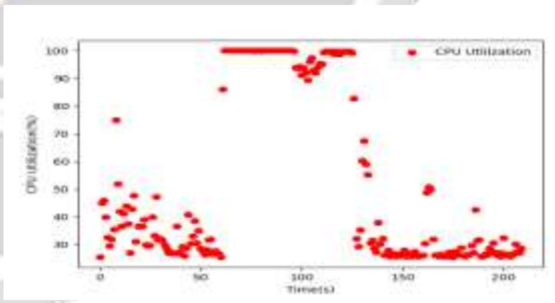

Fig-12: CPU-utilization-nor-attk-counter-measure

Fig.11 illustrates the packet rate under normal conditions and during an attack. It is evident that the implementation of countermeasures resulted in a significant reduction of the packet rate to its normal level.

The diagram fig.12 illustrates the utilization of the CPU by the controller in the presence of both benign and malicious network traffic. A system that manages normal network traffic is likely to exhibit a relatively low CPU utilization rate (30-50%). In the event of a DDoS attack, in which a substantial quantity of malicious traffic is directed at a particular system, CPU utilization increases significantly (80-100%).When the countermeasure is implemented, the CPU utilization rate returns to normal.

6. CONCLUSIONS

SDN is network architecture with the potential to facilitate dynamic and programmable network administration via a centralized controller. Despite this, there are security obstacles, such as DDoS attacks that possess the capability to significantly impede network services and undermine network efficiency. In this study, a dataset was generated using a Mininet and SDN environment. The dataset contains the 25 features enumerated in the table. Machine learning and deep learning models were trained and deployed for classification. The present study introduces a novel approach for detecting DDoS attacks through the utilization of a hybrid deep learning system. The Autoencoder learns to extract useful features from the input data in an unsupervised manner, capturing its inherent patterns and reducing noise. These features are then fed into the DNN, which focuses on the supervised learning task using the extracted features as input. This combination allows for better representation learning and potentially improves the performance. The proposed model was evaluated with other models such as MLP, LSTM and DNN. According to the evaluation results, the model that was proposed exhibited superior performance compared to the other models, achieving a remarkable accuracy rate of 99.98%. Future work entails the development of a ML and deep learning-based detection system capable of promptly and precisely identifying various types of DDOS attacks. Additionally, the system will integrate mitigation tactics and evaluate performance across diverse multi-controller and SDN switch configurations.

6. REFERENCES

[1] Arvind T, K. Radhika, "XGBoost Machine Learning Model-Based DDoS Attack Detection and Mitigation in an SDN Environment," *International Journal of Engineering Trends and Technology*, vol. 71, no. 2, pp. 349-361, 2023. Crossref, https://doi.org/10.14445/22315381/IJETT-V71I2P237

[2] T.V. Phan, M. Park, “Efficient distributed denial-of-service attack defense in SDN-based cloud”, IEEE Access,vol. 7, pp. 18701–18714.,2019.

[3] MS Elsayed, NAL Khac, AD Jurcut,”InSDN: A novel SDN intrusion dataset”, IEEE Access, vol.8, pp. 165263-165284, 2020

[4] T Arvind , Dr.K.Radhika , " Machine Learning Methods for Distributed DoS Attacks: Traffic Generation, Collection and Classification in an SDN Environment" , International Journal of Application or Innovation in Engineering & Management (IJAIEM) , Volume 11, Issue 8, August 2022 , pp. 001-008 , ISSN 2319 – 4847.

[5] H. Polat, O. Polat, and A. Cetin, ‘‘Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models,’’ Sustainability, vol. 12, no. 3, p. 1035, Feb. 2020

[6] MS Elsayed, NAL Khac, MA Azer, AD Jurcut, “A Flow-Based Anomaly Detection Approach With Feature Selection Method Against DDoS Attacks in SDNs”,IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING, VOL. 8, NO. 4, DECEMBER 2022

[7] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho,“Deep recurrent neural network for intrusion detection in SDN-based networks,” in Proc. 4th IEEE Conf. Netw. Softwarization Workshops (NetSoft), 2018, pp. 202–206.

[8] Ahuja N, Singal G, Mukhopadhyay D, Kumar N, “Automated DdoS attack detection in software defined networking”, J. Netw.Comput. Appl. 2021;187:103108.

[9] Jupyter notebook, availableonline: https://jupyter.org/install

[10] Sukhveer Kaur, Krishan Kumar, Naveen Aggarwal, Gurdeep Singh. "A Comprehensive Survey of DDoS Defense Solutions in SDN: Taxonomy, Research Challenges, and Future Directions", Computers & Security, 2021

[11] Novaes, M. P., Carvalho, L. F., Lloret, J., & Proenca, M. L,” Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment”,. *IEEE Access*,vol. *8*, pp. 83765–83781,2020. https://doi.org/10.1109/ACCESS.2020.2992044

[12] T Arvind, and Dr.K.Radhika, “An SDN Based DDoS Traffic Generation, Collection and Classification Using Machine Learning Techniques,” International Conference on Advanced Engineering Optimization Through Intelligent Techniques, Sardar Vallabhbhai National Institute of Technology, 2022.

[13] Hassan A. Alamri, Vijey Thayananthan. "Bandwidth Control Mechanism and Extreme Gradient Boosting Algorithm for Protecting Software-Defined Networks Against DDo S Attacks" , IEEE Access, 2020

[14] RYU SDN Framework Ryubook 1.0 documentation, Retrieved from https://osrg.github.io/ryu-book/en/html.

[15] Ryu Documentation, availableonline: https://ryu.readthedocs.io/en/latest/getting_started.html.

[16] "Introduction to Mininet", GitHub, availableonline: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet [Accessed: March. 05, 2020].

[17] V Deepa , K. Muthamil Sudar, and P Deepalakshmi, "Detection of DDoS Attack on SDN Control Plane using Hybrid Machine Learning Techniques," Proceedings of the International Conference on Smart Systems and Inventive Technology, pp. 299-303, 2018. Crossref, https://doi.org/10.1109/ICSSIT.2018.8748836

[18] Tsung-Han Lee, Lin-Huang Chang, and Chao-Wei Syu, "Deep Learning Enabled Intrusion Detection and Prevention System over SDN Networks," 2020 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1-6, 2020. Crossref, https://doi.org/10.1109/iccworkshops49005.2020.9145085

[19] Afsaneh Banitalebi Dehkordi, Mohammad Reza Soltanaghaei, and Farsad Zamani Boroujeni, " The DDos Attacks Detection through Machine Learning and Statistical Methods in SDN," Journal of Supercomputing, vol. 77, no. 3, pp. 2383–2415, 2021. Crossref, https://doi.org/10.1007/s11227-020-03323-w

[20] Isa M.M., Mhamdi L.Native SDN intrusion detection using machine learning, IEEE Eighth International Conference on Communications and Networking. ComNet, IEEE (2020), pp. 1-7.

[21] Beny Nugraha, and Rathan Narasimha Murthy, "Deep Learning-based Slow DDoS Attack Detection in SDN-based Networks," IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN, pp. 51–56, 2020. Crossref, https://doi.org/10.1109/NFV-SDN50289.2020.9289894

[22] Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K. K. R., & Iqbal, J. (2020). A Deep CNN Ensemble Framework for Efficient DDoS Attack Detection in Software Defined Networks. *IEEE Access*, *8*, 53972–53983. https://doi.org/10.1109/ACCESS.2020.2976908.

[23] T. M. Nam, P. H. Phong, T. D. Khoa et al., "Self-organizing map-based approaches in DDoS flooding detection using SDN," in *Proceedings of the 2018 International Conference on Information Networking (ICOIN)*, pp. 249–254, IEEE, Chiang Mai, Thailand, January 2018.