Mathematical Statistician and Engineering Applications
ISSN: 2094-0343
2326-9865

An Efficient DDoS Attack Detection Mechanism in SDN Environment

Vanlalruata Hnamte[*], **Jamal Hussain**
Department of Mathematics and Computer Science,
Mizoram University, Mizoram, India
Corresponding author's email: vanlalruata.hnamte@gmail.com

Article Info
Page Number: 573-592
Publication Issue:
Vol. 72 No. 1 (2023)

Article History
Article Received: *15 October 2022*
Revised: *24 November 2022*
Accepted: *18 December 2022*

Abstract
As cyberattacks get more sophisticated, it becomes more difficult to identify advanced attacks in a wide variety of industries, including business, national defense, and healthcare. Traditional intrusion detection systems are insufficient to identify these sophisticated attempts with unpredictable patterns. Attackers evade recognized signatures and spoof legitimate users. While Software-Defined Network (SDN) provides greater innovation to the design of future networks, it is also more susceptible to network attacks. We present a system for the detection and protection of network attacks in the SDN environment using Deep Learning (DL) for addressing challenges. We use InSDN, the most recent IDS dataset specially developed for the SDN environment. InSDN includes sophisticated DoS and network attacks and, it is a publicly accessible dataset. Additionally, we compare the model performance trained using CIC-IDS2017 and CIC-DDoS2019 datasets respectively. We focus on the DDoS attack category in this study and create an intrusion detection DL model for SDN. We construct our model using a Convolutional Neural Network (CNN) and assess its performance. Additionally, we propose the optimized CNN design for improved performance based on various evaluations.
Keywords: - CIC-IDS2017, DCNN, SDN, IDS, CIC-DDoS2019, InSDN.

Introduction

In the past few years, security, data integrity, and privacy have become more important parts of modern communication networks. Organizations are required to protect their sensitive data from both internal and external threats. Some authorized users may send their company's data to other parties for a variety of reasons. A constant live data flow makes it harder to detect a real-time attack. Advances in networks consist of hardware or software or both, entities that are customized to match the needs of firms in a seemingly random way. These elements include threats, vulnerabilities, and security restrictions. The program employs complex attacks that put the data in great danger. Using log files, the developer and programmer may assure the security of the system. The complexity of contemporary communication networks renders systems susceptible to security breaches. Many types of network attacks vary in speed and power depending on how the network is set up. The primary problem in current network security is identifying and neutralizing attacks promptly and in real time. Therefore, Intrusion Detection System (IDS) technology is required to detect all internal and external attackers and secure systems and network architecture.

In the past few years, the SDN paradigm has become a new way to design networks to meet the needs of new applications. SDN offers resources for enhancing network control and administration by separating the control and data planes and centralizing the logical control via a controller [1]. SDN allows the network to be managed by software that permits the use of less networking equipment and simplifies physical connection and setup. Thus, network operators can change network behavior to accommodate current services and security applications. Therefore, the bulk of network services will be more adaptable, programmable, and platform-independent [2]. However, the centralized control logic might be a prime target for malicious attacks, namely Distributed Denial of Service network operations [3], [4].

DL technology can readily extract and learn features and patterns from the network's behavior, hence offering important insights for attack detection [5]. Furthermore, as the number of users increases, network traffic becomes more complicated, and deep neural networks are effective at reducing network traffic complexity analysis due to their ability to learn enormously complex data representations and manage them without human intervention [6].

This study suggests an IDS built for SDNs, termed Deep CNN (DCNN). DCNN is a novel technique that can identify multiple attacks on the SDN for real-time usage. CNN model is one of the DL algorithms used in picture categorization, processing text, and documents, Bio-metric Recognition, etc., [4]. The proposed method reduces training time while maintaining accurate detection in the current IDS. Therefore, the suggested model enables a lighter mechanism for SDN while ensuring that the performance achieves good accuracy and perfection detection.

Recent Studies

Scott-Hayward et al. [7] stated that significant efforts have been committed to resolving security vulnerabilities and providing mitigating measures in response to the rising deployment of SDN technology.

Dridi et al. [8] have proposed the SDN-Guard solution. It involves protecting SDN networks from DoS attacks by rerouting traffic that could be harmful, changing flow timeouts, and combining flow rules. The approach uses an IDS to figure out how likely it is that each new flow will pose a threat. However, this technique adds communication overhead to the control plane owing to the exchange of messages between the controller, the IDS, and network functionalities deployed within the cloud [9].

Niyaz et al. [10] proposed an IDS based on DL that identifies network attacks on an SDN for the detection of multiple kinds of attacks. The study used TCFI, FE, and TC which are the three modules discussed. The system looks at the packets in the SDN controller, pulls out the features, and then sends the features to be labeled as normal or malicious. The tcpdump and hping3 tools were used to produce network traffic. In the FE module, the proposed system harvests 68 characteristics for each packet's categorization. However, it was only accurate 95.65% of the time. Since this operation is done on each packet, the number of extracted characteristics requires a lot of memory and a long time to process. In addition, the majority of these retrieved characteristics are unrelated to network attack techniques. The authors also broke the rules of SDN design by setting up the controller so that all packets had to go through

the controller to be handled and by not using the flow table function. When such a model is run on large networks, the controller will crash, which means that small networks won't work well.

Hussain et al. [11] suggested a detection strategy by applying a Deep Neural Network (DNN) approach to recognize the network attacks in the SDN networks. The KDD-CUP99, NSL-KDD, and UNSW-NB15 public datasets were utilized throughout the training and testing phases. The suggested model used the basic fundamental characteristics for classification. However, due to the limited number of features, the accuracy of the detection stage was only 99.61%. These characteristics have been collected from the fundamental data of the flow and are insufficient to cover the attack behaviors; hence, the attacker may easily circumvent the IDS.

Wang et al. [12] developed a network IDS using a hybrid neural network comprised of flexible transformers and a CNN algorithm. The suggested model was able to make accurate predictions 99.86% of the time, which is pretty high. Combining two or more models usually increases the model's complexity, requiring more memory and CPU processing power [12]. Their proposed IDS is only capable of detecting network attacks. The model that was suggested has a lot of extracted characteristics, which require more memory and a longer process. Therefore, it will create a bottleneck at the controller and cannot be applied in large and complex networks for real-time usage. Most of these traits that were found have nothing to do with how network attacks work.

A model was provided by Choobdar et al. [13] which suggested DL for the SDN environment to recognize network attacks within the controller and hosts. Utilizing the usual DL method with the Relu and Softmax functions, the model described in this study identifies the attack. During the training and testing stages, the CICIDS2017 public dataset was utilized. The suggested model produced a prediction accuracy of 99.6%, which is likewise regarded as being rather excellent. Due to a large number of retrieved characteristics, however, the proposed model needs a lot of memory and takes a long time to run on the CPU. Some of these functions are inaccessible in the SDN environment or need extra processes, resulting in OpenFlow channel congestion and controller overhead.

Tang et al. [14] developed a lightweight solution to minimize controller overload. This method classifies traffic in real-time using a Gated Recurrent Unit Recurrent Neural Network (GRU-RNN). During training and testing, the NSL-KDD public dataset was used. The suggested model used six fundamental characteristics for categorization. These characteristics are retrieved in real-time for each flow inside the SDN controller. The provided model, on the other hand, had a very low detection rate i.e. 89% accuracy rate. Because it is simple and doesn't cover real-time attacks well, and because it only uses a small number of retrieved characteristics, it isn't good at detecting threats that can quickly disrupt network resources.

Abubakar et al. [15] developed a method for identifying attacks using the neural network technique used to categorize flow in an SDN environment. The publicly available NSL-KDD dataset is used to train and validate their proposal system. The objective of the model is to define DoS, U2R, R2L, and Probes. The detection rate of the model was 97.4%. In the training phase, however, using a small dataset has a big effect on how well the real test works. The only thing used to get the information is the packet header, which doesn't take into account

attack patterns. The controller had to handle each packet, which caused a bottleneck and caused the controller to fail. Also, the suggested model needs a way to choose the effective features in case of an attack.

The key challenges with DL methodologies are feature extraction and selection methods [16]. Existing systems suffer from a large number of categorization characteristics or use just fundamental aspects. Utilizing several functions generates network congestion and undesirable latency. While employing a limited number/of basic characteristics does not enable reliable detection of attacks since it does not account for attack behavior [17].

The majority of studies from Table 1 suggest models that need real-time monitoring of traffic; hence, this procedure must monitor and check each packet or flow traversing and categorize it as benign or malicious using a particular model. To gather, analyze, and categorize the traffic, processing demands additional time, memory, and CPU consumption. This procedure congests the controller and switches. Researchers must thus address the tradeoff between classifier efficiency and network performance.

Methodology

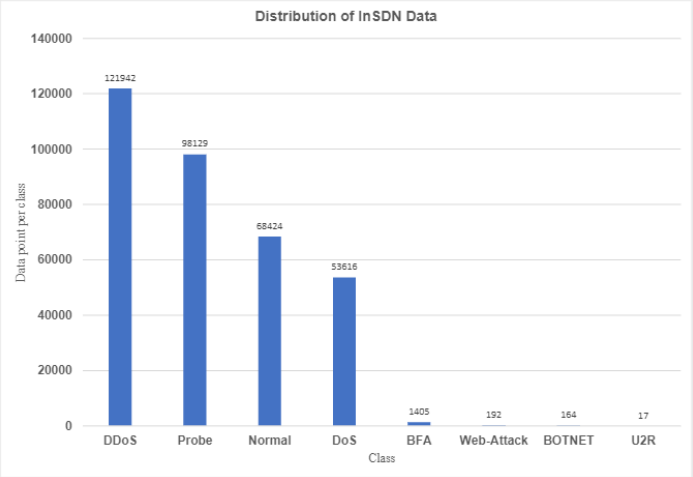
We provide the suggested SDN protection mechanism in this section. The advantages and easier operation of SDN may be attributed to its central administration and programmability. This study proposes a methodology for detecting DDoS attacks using SDN anomaly detection.

Table 1. Recent Studies

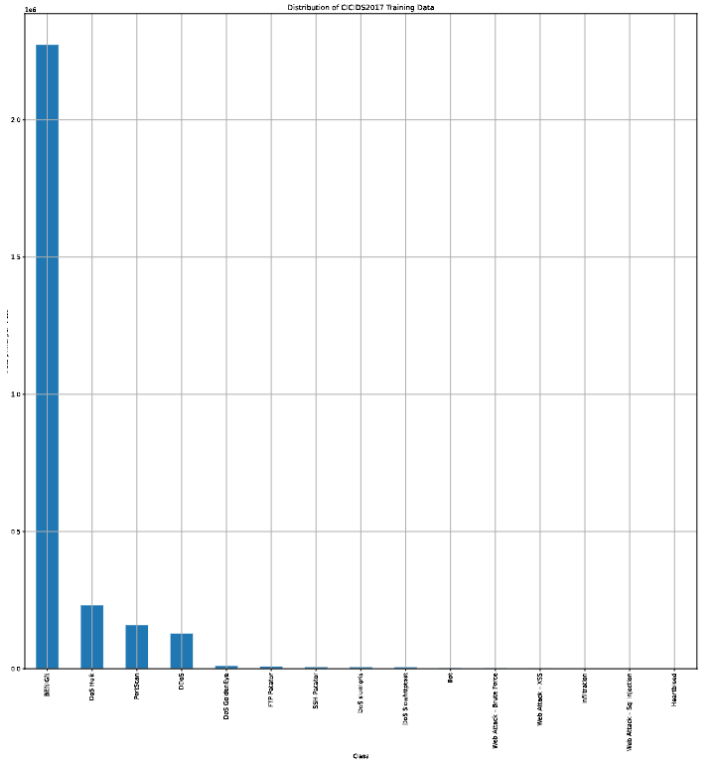
Sl No	Authors	Dataset	Model	Accuracy	Loss
1	Niyaz et al. [10]	KDDCUP99	SAE	95.68%	0.5
2	Hussain et al. [11]	KDDCUP99 NSL-KDD UNSW-NB15	DNN	99.69% 92.12% 81.70%	0.0207 0.1615 0.5245
3	Wang et al. [12]	CICDDoS2019	DDoSTC	99.90%	*
4	Choobdar et al. [13]	NSL-KDD CICIDS2017	SAE	98.5%	*
5	Tang et al. [14]	NSL-KDD	GRU-RNN	89%	*
6	Abubakar et al. [15]	NSL-KDD	BP	97%	*
7	***This Paper***	InSDN CICIDS2017 CICDDoS2019	DCNN	99.99% 99.97% 99.77%	0.0010 0.0010 0.0069

We utilize three datasets; the InSDN dataset, CIC-IDS2017 dataset, and CIC-DDoS2019 dataset, for training the detection model for real-time network attacks and to validate the network protection architecture as it contains DDoS attack as shown in Fig. 1. The training of a deep computation model is difficult due to the enormous number of parameters normally included in such a model. Specifically, it requires a high-performance computer with a huge memory and a strong processing unit. Our DL model training experiment is done on the system with an Intel i9 10900k CPU, NVIDIA RTX 3060Ti GPU, and 64 GB of RAM in the hardware environment, and the Ubuntu 18.04.6 operating system with Keras and TensorFlow

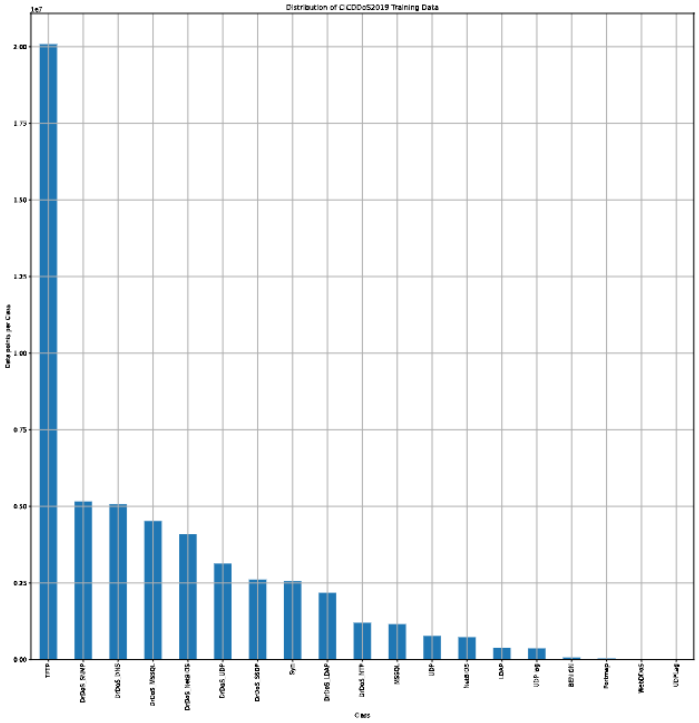
installed in the system software. The system for detecting mechanisms against network attacks in real-time is constructed using the same ubuntu operating system. The flow list is sent to the OpenFlow switch through the OpenVSwitch command.



(a) InSDN Data Distribution



(b) CICIDS2017 Data Distribution



(c) CICDDoS2019 Data Distribution

Figure 1. Dataset Data Distribution

The proposed SDN protection system we provide here is split into two parts: detection and defense. To facilitate logical communication, they are physically separated on the controller, as illustrated in Fig. 2. Sub-modules make up half of each module. An alert is sent out by the SDN defense system whenever a DDoS attack is detected, and once the controller gets this anomaly detection alert, it immediately triggers the mitigation module to take action. The detection module has the following two modules: the flow or packet gathering module, which is responsible for gathering incoming packets and for flow rules, and the abnormal detection module, which is responsible for detecting and processing the gathered data. There are two sub-modules inside the mitigation module as well: the IP traceback module, which is responsible for tracking down the source of the abnormal flow, and the defense mechanism module, which is responsible for containing the abnormal traffic and reducing the severity of the DDoS attack.

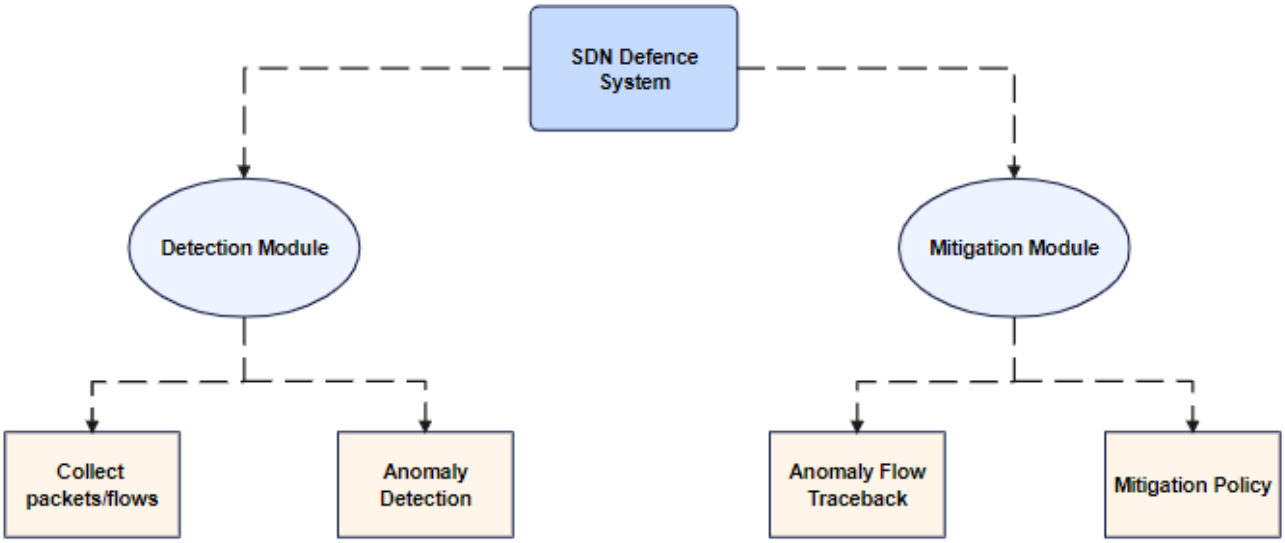


Figure 2. SDN DDoS Detection and Prevention

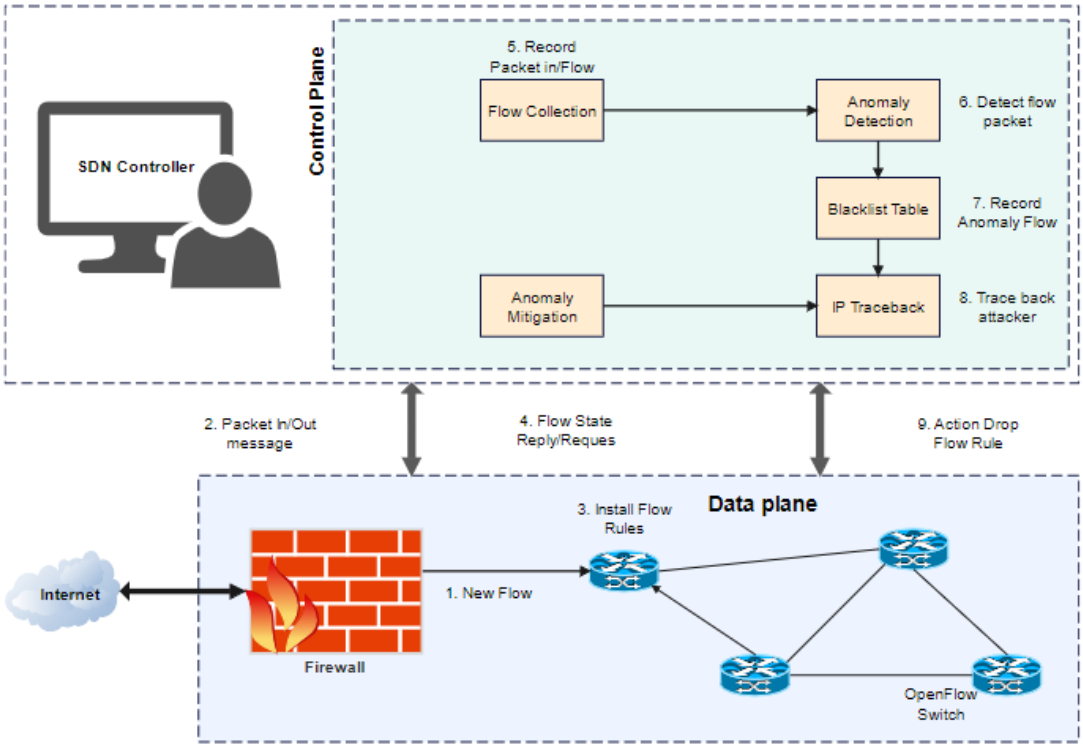


Figure 2. SDN Defence System

In Fig. 3, we can see the detailed process that takes place when the SDN defense system is activated on the SDN controller. The SDN controller will use the ofp packet in command to parse the packet's header feature fields and forward them to the controller's exception detection module. A value of 0 indicates that the packet is considered normal, whereas a value of 1 indicates that the packet is considered abnormal and its details are sent to the

blacklist table. Furthermore, the controller uses the ofp flow command to periodically request the switch's flow table data. The acquired flow table data is then transferred from the controller to the abnormal detection module, which uses the representation of normal as a value of 0 to perform normal forwarding. When there is an abnormal amount of traffic, a 1 is recorded and submitted to the blacklist table. Finally, to mitigate the effects of DDoS attacks in SDN, we employ an IP traceback technique to look up abnormal flow data in the blacklist table, identify the attacker sources, and then trigger flow rules to the controller via ofp flow message to hold back the source port which is under attack. Algorithm 1 depicts the process flow.

Algorithm 1 SDN Detection System

Input Trafic Sequences
Output Send to Defence Module

```
If ofp_packet_in then
Packet(Features) = feature_extration [ofp_packet_in]
Result=DCNN(Packet(Features))
If Result == 0 then
Send Packet_Out Message
else
Send Log to Blacklists Table
End if
End if
If ofp_flow_states == reply then
Packet(Features) = feature_extration(ofp_flow_stat)
If reply == 0 then
Forward Packet
Else
Send Log to Blacklists Table
End if
End if
IP traceback(Blacklists)
Action(Block_Attack_Port)
```

The detection accuracy of a model trained using machine learning for classification relies heavily on the quality of the features used in the model, making it imperative to use sophisticated mathematical and empirical reasoning in the process of feature extraction. On the other hand, DL allows the model to automatically extract information from several layers, integrate them, and then use the combined outputs to make predictions.

CNN uses a series of convolutional and pooling layers built on top of one another to extract characteristics from objects. However, CNN's pooling layers are not an effective classifier for IDS data since they are employed to identify mostly for image classification, minimizing global error, resulting in a lengthy training process and poor network generalization.

In this study, the DCNN model is utilized for detecting network attacks. DCNN is a model for IDS in the SDN environment. The system model is depicted in Fig. 4. The pooling layer is eliminated to decrease the workload of the model as the proposed model is not for processing any image, as the main purpose of this layer is to reduce the size of feature maps. The ReLU

activation function is used to activate each hidden layer. ReLU equation is defined in Equation (1). Only with the output layer, the softmax activation function is utilized. The softmax activation function is defined in Equation (2).

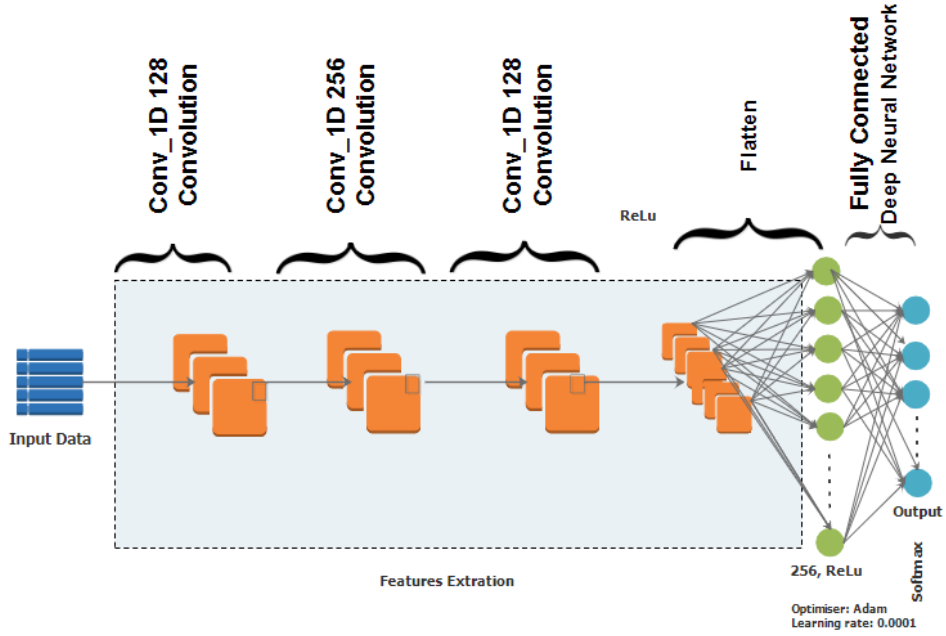


Figure 4. CNN Architecture

For the optimization, adam is used. Adam is an adaptive learning rate method, which calculates individual learning rates for various parameters. The learning rate for each weight is defined as 0.0001.

$$ReLu = max(0, x) \tag{1}$$

The use of ReLU activation aids in stopping the computation needed to run the neural network from growing exponentially. The computational cost of including more ReLUs grows linearly with the size of the CNN.

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{2}$$

The Softmax Function often has connections to the Cross-Entropy Function. After using the Softmax Function in CNN, the next step in optimizing the network's performance involves testing the model's robustness with the Cross-Entropy Function as the Loss Function.

A. Dataset and feature dropping

The InSDN dataset [18] was used as one of the three datasets to train DCNN. The InSDN is a new dataset established in September 2020 by the University College Dublin team. This dataset was created through a plausible technique. Most public dataset concerns, such as redundancy, are resolved by the InSDN [18].

Table 2 displays the properties of the automatically generated flow table which are also available in InSDN Dataset [18].

Table 2. Flow Table Data

Src_ip	Dst_ip
Src_mac	Dst_mac
In-port	Src_port
Dst_port	Protocol
Duration	ByteCount
PacketCount	

The InSDN dataset is the first publicly available dataset to be developed under the SDN environment [18]. In addition, it contains updated types of attacks and mimics novel patterns. DoS, network, Probe, U2R, Web-Attack, BotNET, and Brute Force Attacks are included under the InSDN. Table 3 represents the features that were dropped from the original features from InSDN dataset for training the proposed model.

Table 3. InSDN Dataset Dropped Features

Features	Description
Flow ID	Unique Flow Identification allotted
Src IP	The source IP Address
Src Port	The source port number
Dst IP	The destination IP Address
Dst Port	The destination port number
Timestamp	The accessing date and time

Features removed from the InSDN dataset, shown in Table 3 are to identify abnormal traffic associated with DDoS attacks as part of the model's input feature dataset in order to increase the model's detection accuracy and verify the credibility of the findings.

The CIC-IDS2017 dataset [19] was one of the three datasets used to train the DCNN. Due to the fact that it includes dangers not covered by previous databases, this one has attracted the interest of many scientists. While doing an experimental study on CICIDS2017, it became clear that there were a number of significant flaws in the dataset [20]. The CICIDS2017 has been upgraded to include the latest prevalent threats including DoS, DDoS, brute force, cross-site scripting (XSS), SQL injection, botnet, and port scanning [19] There was a 5-day data collection session beginning at 9 a.m. on Monday, July 3, and ending at 5 p.m. on Friday, July 7, 2017. There will only be normal traffic on Mondays. With over 80 network traffic characteristics collected and computed using the CICFlowMeter open-source software, the CICIDS2017 is fully labeled and accessible on the Canadian Institute for Cyber Security's website for anyone to use. Table 4 represents the features that were dropped from the original features from the CIC-IDS2017 dataset for training the proposed model.

Table 4. CICIDS2017 Dataset Dropped Features

Features	Description
Flow ID	Unique Flow Identification allotted
Source IP	The source IP Address
Destination IP	The destination IP Address
Timestamp	The accessing date and time

The CIC-DDoS2019 dataset [21] was one of the three datasets used to train the DCNN. The CIC-DDoS2019 dataset is a simulation of real-world data, consisting of benign and modern instances of popular DDoS attacks. The CICFlowMeter-V3 network traffic analyzer was used, to generate labeled flows that contain timestamps, IP addresses, ports, protocols, and attack vectors [21]. The data set has been provided on a day-by-day basis. Daily raw data was captured, consisting of Windows and Ubuntu system event logs using the pcaps tool. Table 5 represents the features that were dropped from the original features from the CIC-DDoS2019 dataset for training the proposed model.

B. Model training and attack evaluation

We use the network configuration shown in Fig. 5 to do simulation experiments that highlight the IP traceback procedure. There are three controllers (C1, C2, and C3) and four hosts (h1, h2, h3, and h4) in this architecture. The attacking hosts, h1 and h2, are contrasted with the non-threatening hosts, h3 and h4, which are used by regular users. Hosts h1 and h2 initiate a distributed denial-of-service (DDoS) attack against host h4.

Table 5. CICDDoS2019 Dataset Dropped Features

Features	Description
Flow ID	Unique Flow Identification allotted
Source IP	The source IP Address
Destination IP	The destination IP Address
Timestamp	The accessing date and time
SimilarHTTP	HTTP Related Protocols
Fwd URG Flags	Forward urgent Flags
Bwd URG Flags	Backward urgent Flags
FIN Flag Count	FIN Flag Counting
PSH Flag Count	PSH Flag Counting
ECE Flag Count	ECE Flag Counting
Fwd Avg Bytes/Bulk	Average Forward Bytes
Fwd Avg Packets/Bulk	Average Forward Packets
Fwd Avg Bulk Rate	Average Forward Bulk Rate
Bwd Avg Bytes/Bulk	Average Backward Byte
Bwd Avg Packets/Bulk	Average Backward Packets
Bwd Avg Bulk Rate	Average Backward Bulk Rate

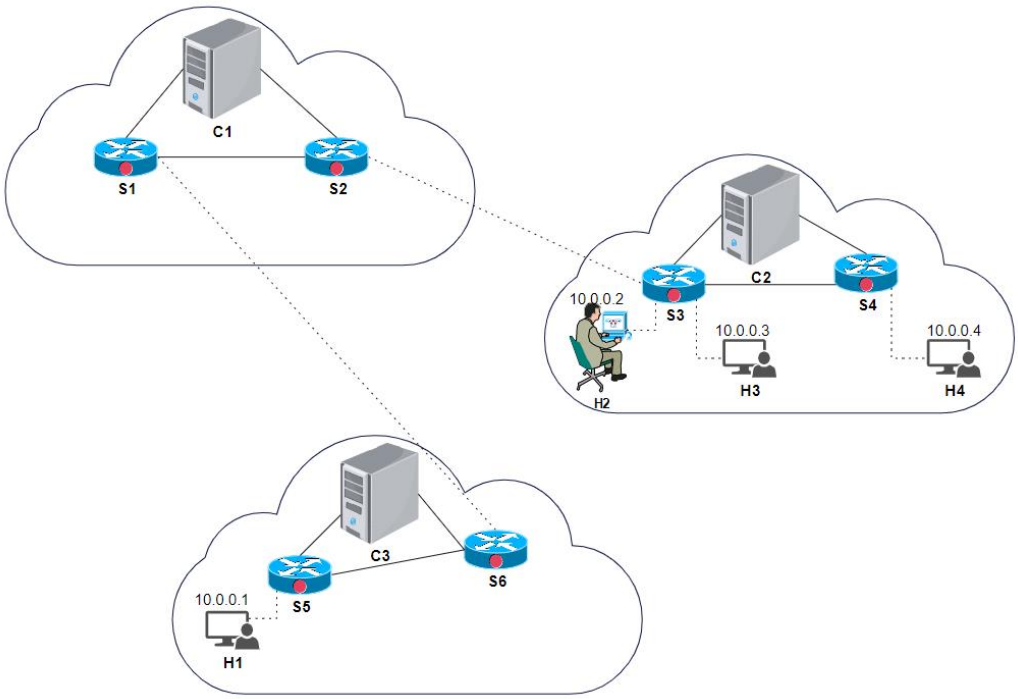


Figure 5. SDN Topology Model

CNN is an artificial feedforward neural network. The CNN is then connected to a deep neural network consisting of three hidden layers, then to the output layers. The size of the data features may be decreased by pooling and assigning feature constantly, but the proposed model eliminates the pooling to reduce the training time. CNN can thereby ensure the reliability of the input features data.

The preceding layer's feature is convolved using a convolution kernel in the convolution layer. The output of the activation function is given as a characteristic of the current layer x_j^l, as given in Equation (3).

$$X_j^l = f\left(\sum_{i \in M_j} x_j^{l-1} * k_{ij}^l + b_j^l\right) \tag{3}$$

where l represents the number of network layers, k represents the instances of convolution kernels, M_j represents the instances of data features, and b represents the offset. Theoretically, Equation (4) explains when sampling the data, the number of features after sampling in the subsampling layer is still the same:

$$Y_j^l = f\left(\beta_j down\left(x_l^{l-1}\right) + b_j^l\right) \tag{4}$$

where β is the weight, $down(*)$ is the subsampling function, and $*$ is the subsampling function's input data.

Result and Discussion

Through topology and tracing, we can identify that the attack originated from switch namely S3 in control domain C2 and switch S5 in control domain C3, respectively. We can issue or trigger flow rules recorded in the blacklist table to the attack source switch within or from a particular control domain in an effort to halt the attack and monitor the victim host's traffic trend changes. Fig. 6 shows that the attacker begins attacking h3 at 05:21:50 IST, and that the quantity of h3 packets begins to decrease and traffic gradually returns to normal beginning at 05:22:55 IST, suggesting that the defensive action was effective.

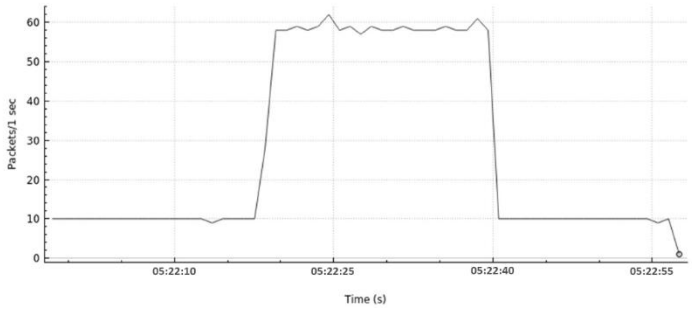


Figure 6. Flow Record after attack defence

The performance of the approach in our model were evaluated and tested using the following metrics as given by [22].

- False Positive (FP): A quantity of normal traffic is wrongly identified as attack traffic.
- False Negative (FN): The amount of attacks traffic is incorrectly categorised as normal traffic.
- True Positive (TP): The proportion of attacks traffic successfully identified as attack traffic.
- True Negative (TN): The number of regular traffic that is appropriately identified as normal traffic.
- Accuracy (AC): The number of correct forecasts relative to total traffic. This accuracy has indeed been calculated using the following equation:

$$AC = \frac{TP+TN}{TP+TN+FP+FN} * 100\% \quad (5)$$

- Precision (P): commonly known as a false alarm, it is used to figure out how much of attack traffic accurately identified. The calculation is equation is given below:

$$P = \frac{TP}{TP+FP} * 100\% \quad (6)$$

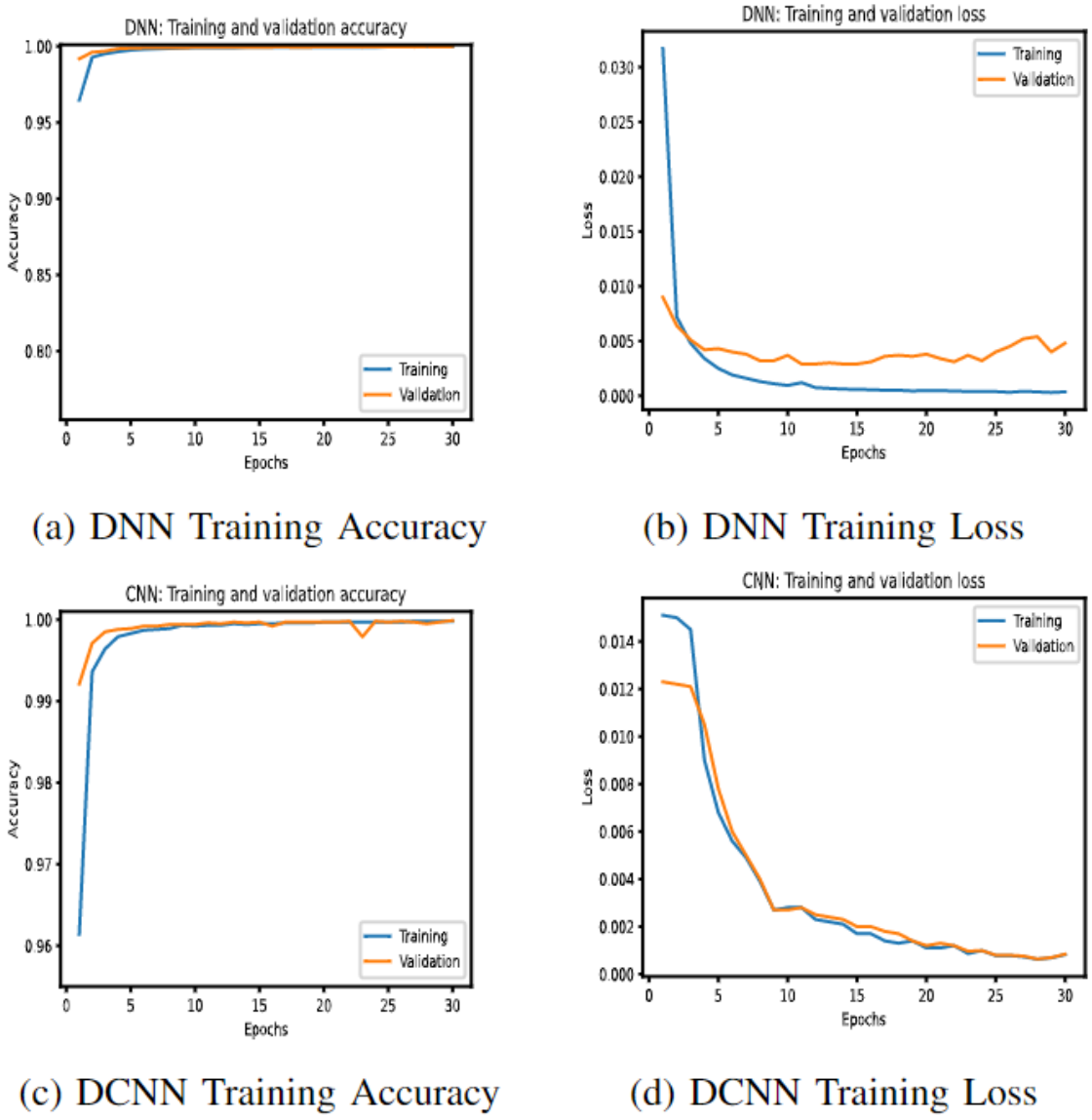
- Recall (R): measures the rate of anticipated attacks relative to overall attack traffic. The calculation is determined by the formula:

$$R = \frac{TP}{TP+FN} * 100\% \quad (7)$$

- F1-measure (F1): provides a more accurate estimate of the model's precision based on both R and P as given below:

$$F1 = \frac{2}{\frac{1}{P}+\frac{1}{R}} * 100\% \quad (8)$$

Using the above-mentioned metrics on a multiclass-labeled dataset, the DCNN was evaluated. To categorize sample flows in real time, the CNN algorithm has been employed. The DCNN design and requirements are shown in Fig. 4 and Table 6. Experiments were done by training and evaluating a model firstly, with 78 features from the InSDN dataset, and the outcome has been illustrated in Fig. 7 and Fig. 8.

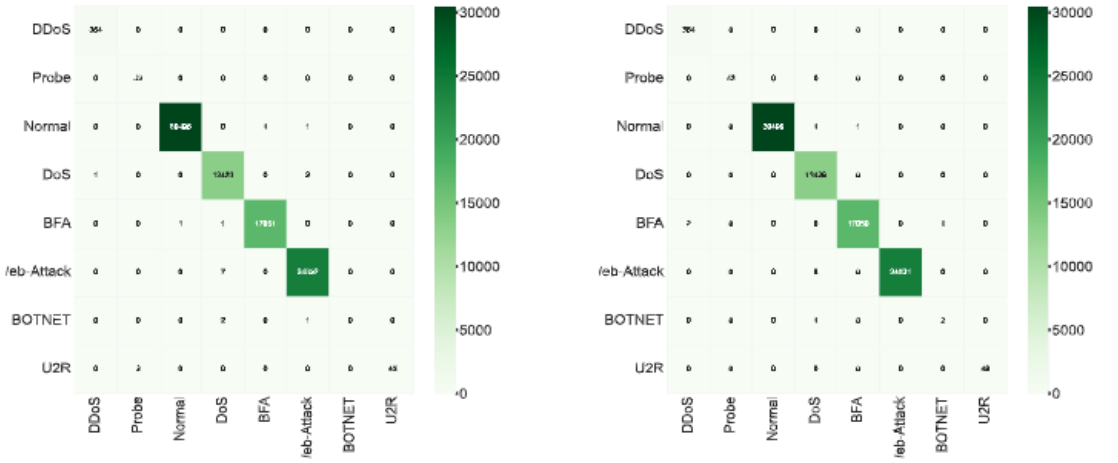


(a) DNN Training Accuracy (b) DNN Training Loss

(c) DCNN Training Accuracy (d) DCNN Training Loss

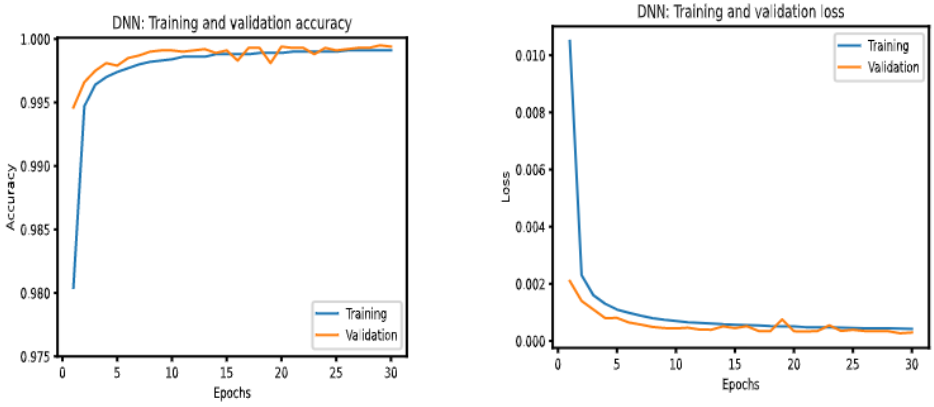
Figure 7. Training Validation and Loss: InSDN

The second experiments were done by training and evaluating a model with 81 features from the CICIDS2017 Dataset, and the outcome has been illustrated in Fig. 9 and Fig. 10.

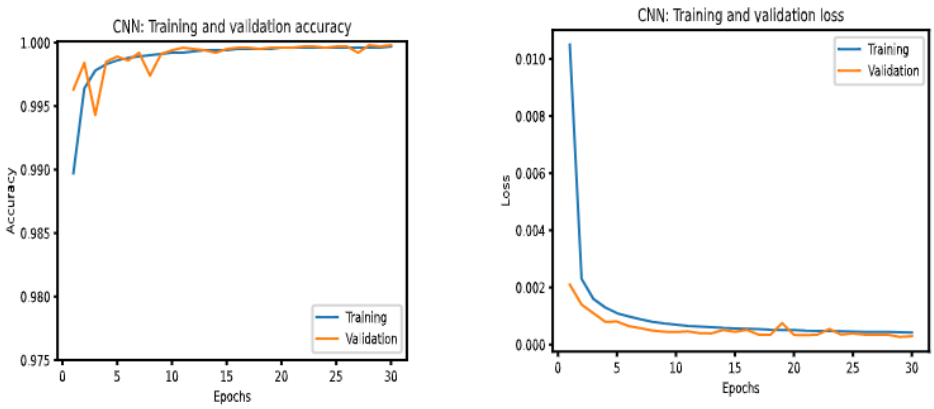


(a) DNN Confusion Matrix (b) DCNN Confusion Matrix

Figure 8. Confusion Matrix

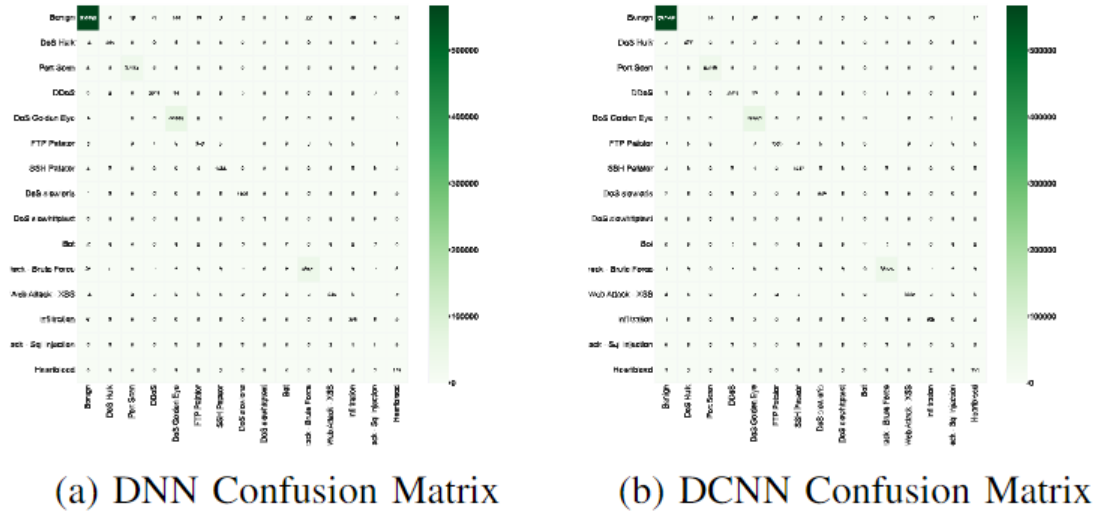


(a) DNN Training Accuracy (b) DNN Training Loss



(c) DCNN Training Accuracy (d) DCNN Training Loss

Figure 9. Training Validation and Loss: CICIDS2017



(a) DNN Confusion Matrix (b) DCNN Confusion Matrix

Figure 10. Confusion Matrix

The third experiments were done by training and evaluating a model with 67 features from the CIC-DDoS2019 Dataset, and the outcome has been illustrated in Fig. 11 and Fig. 12.

The comparison of the outcomes of the studies is shown in Table 7. Our DCNN model scored great on P, R, F1, and accuracy.

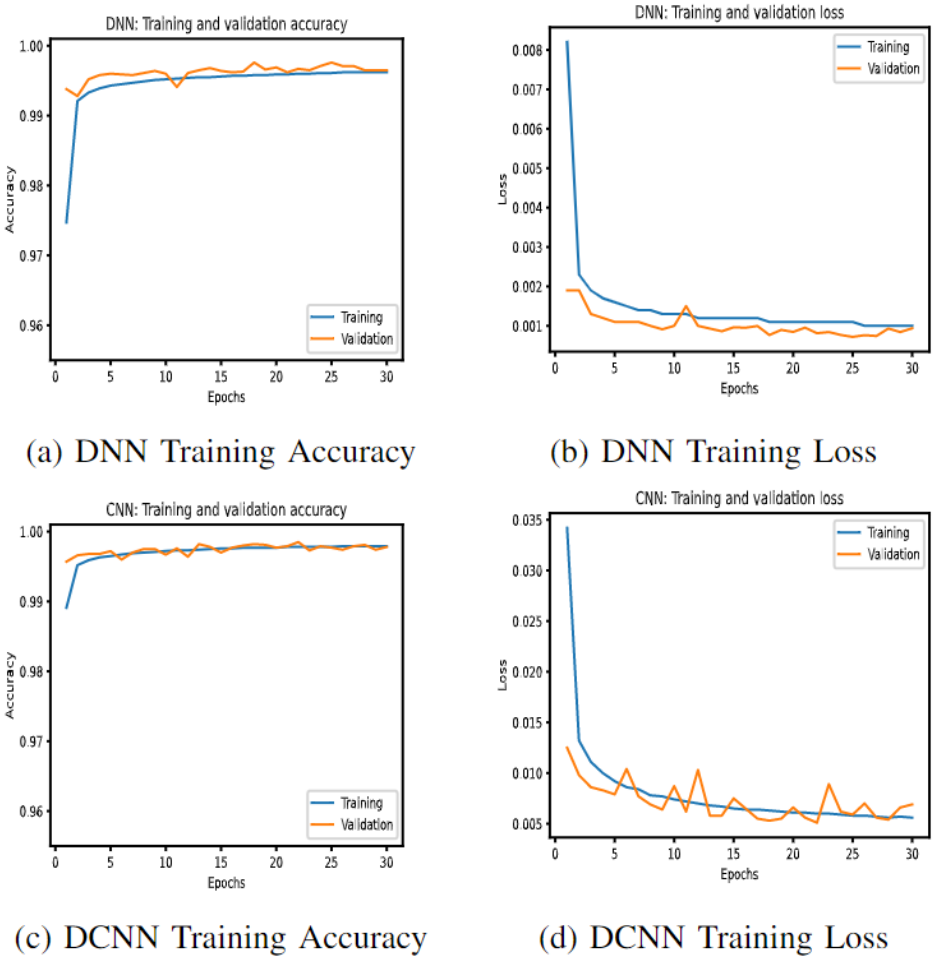
The suggested model has achieved 99.99% accuracy when trained with the InSDN dataset, 99.97% accuracy when trained with CIC-IDS2017 dataset, and 99.77% accuracy when trained with CIC-DDoS2019 dataset. This demonstrates that the suggested approach is effective and capable of detecting attacks with very high precision.

Conclusion

In this study, we present a network attack detection mechanism approach based on DL and mitigation approach; the model training accuracy of the network attack using the InSDN dataset achieved 99.99\%. Furthermore, we describe the design approach for each function module, define the associated functions, and simulate SDN traffic in an SDN environment. Realtime attacks are used to assess the defensive effectiveness of the defense architecture. According to the final testing findings, the attack detection technique based on DL is essentially usable. We can also acquire the proper detection results based on additional feature fields of attack data traffic such as timespan, request frequency, and other fields. OpenFlow flow entries created as a consequence of attack detection may efficiently clean attack traffic, hence mitigating attacks in SDN networks. The efficacy of the network attack defense strategy based on DL in the SDN network for identifying and defending against network attacks is finally validated in a real-time network environment.

This experiment demonstrates the network attack detection mechanism based on DL achieves high detection accuracy, minimal dependence on the system or application, and flexibility in updating the network model, thereby compensating for the deficiencies of the existing network attack detection mechanism. The benefits of a proposed network attack detection mechanism based on DL can be summed up as: it improves the accuracy of network attack detection, but it also reduces the training time, suggests the action of the detection

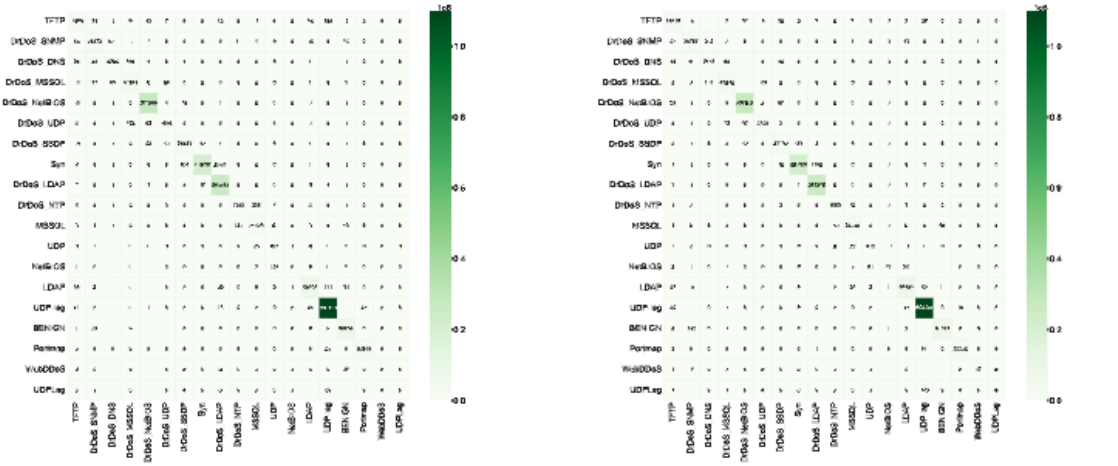
system if the abnormal flow is detected on a network, and the proposed mechanism will still be applicable for all network model even in future without extra efforts.



(a) DNN Training Accuracy (b) DNN Training Loss

(c) DCNN Training Accuracy (d) DCNN Training Loss

Figure 11. Training Validation and Loss: CICDDoS2019



(a) DNN Confusion Matrix (b) DCNN Confusion Matrix

Figure 12. Confusion Matrix

Table 6. The model description

Variable	Parameters
Convolutional1D	3 Layers
Activation Function	ReLU
Optimizer Function	Adam with Learning Rate 0.0001
Loss Function	Categorical Crossentropy
Flatten	4391168
Dense	32896
Dropout	0.1
Dense	12384
Dropout	0.1
Dense	12416
Dropout	0.1
Dense	12384
Dropout	0.1
Dense	776
Output	8
Epoch	30
Kernel	Size 6
Batch	Size 64

This experiment demonstrates the network attack detection mechanism based on DL achieves high detection accuracy, minimal dependence on the system or application, and flexibility in updating the network model, thereby compensating for the deficiencies of the existing network attack detection mechanism. The benefits of a proposed network attack detection mechanism based on DL can be summed up as: it improves the accuracy of network attack detection, but it also reduces the training time, suggests the action of the detection system if the abnormal flow is detected on a network, and the proposed mechanism will still be applicable for all network model even in future without extra efforts.

To make use of the benefits of a CNN-based classifier, the model has been optimized to perform faster. This strategy improved the CNN predictor's accuracy and decreased training time. Since the SDN controller is centrally located, it is a prime target for DDoS attacks. This kind of attack on the controller is now the most difficult challenge in SDN. In the future, we want to conduct a more extensive evaluation of the suggested approach for a variety of DDoS attacks in SDN and will examine how well the method performs in terms of network size and on several types of networks.

Table 7. Evaluation Performance Comparison

Dataset	Model	Accuracy	Loss	Precision	Recall	F1-Score
InSDN	DNN	99.97	0.0048	99.97	99.97	99.97
	DCNN	99.99	0.0010	99.99	99.99	99.99
CICIDS2017	DNN	99.90	0.0044	99.90	99.90	99.90

	DCNN	99.97	0.0010	99.97	99.97	99.97
CICDDoS2019	DNN	99.65	0.0093	99.65	99.94	99.64
	DCNN	99.77	0.0069	99.77	99.77	99.77

References

[1] M.P. Novaes, L.F. Carvalho, J. Lloret, M.L. Proenca, "Adversarial deep learning approach detection and defense against ddos attacks in sdn environments," Future Generation Computer Systems, vol. 125, Dec. 2021, pp. 156–167, doi:10.1016/j.future.2021.06.047.

[2] J. Kim, J. Kim, H. Kim, M. Shim, E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," Electronics, vol. 9(6), Jun. 2020, doi:10.3390/electronics9060916.

[3] T. Das, V. Sridharan, M. Gurusamy, "A survey on controller placement in SDN," IEEE Communications Surveys & Tutorials, vol. 22(1), Aug. 2019, pp. 472–503, doi:10.1109/COMST.2019.2935453.

[4] L. Alzubaidi, J. Zhang, A.J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamarıa, M.A. Fadhel, M. Al-Amidie, L. Farhan, "Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions," Journal of Big Data, vol. 8(1), Mar. 2021, pp. 1–74, doi:10.1186/s40537-021-00444-8.

[5] J. Hussain, V. Hnamte, "Deep learning based intrusion detection system: Software defined network," Proc. 2021 Asian Conference on Innovation in Technology (ASIANCON), IEEE, Oct. 2021, pp. 1–6, doi:10.1109/ASIANCON51346.2021.9544913.

[6] W.G. Hatcher, W. Yu, "A survey of deep learning: Platforms, applications and emerging research trends," IEEE Access, vol. 6, Apr. 2018, pp. 24411–24432, doi:10.1109/ACCESS.2018.2830661.

[7] J.C.C. Chica, J.C. Imbachi, J.F.B. Vega, "Security in sdn: A comprehensive survey," Journal of Network and Computer Applications, vol. 159, Jun. 2020, pp. 102595, doi:10.1016/j.jnca.2020.102595.

[8] L. Dridi, M.F. Zhani, "SDN-guard: DoS attacks mitigation in SDN networks," Proc. 2016 5th IEEE International Conference on Cloud Networking (Cloudnet), IEEE, Dec. 2016, pp. 212–217, doi:10.1109/CloudNet.2016.9.

[9] K. Bhushan, B.B. Gupta, "Distributed Denial of Service (DDoS) attack mitigation in Software Defined Network (SDN)-based cloud computing environment," Journal of Ambient Intelligence and Humanized Computing, vol. 10(5), Apr. 2018, pp. 1985–1997, doi:10.1007/s12652-018-0800-9.

[10] Q. Niyaz, W. Sun, A.Y. Javaid, "A deep learning based ddos detection system in Software-Defined Networking (SDN)," EAI Endorsed Transactions on Security and Safety, vol. 4(12), Dec 2017, doi:10.4108/eai.28-12-2017.153515.

[11] J. Hussain, V. Hnamte, "A novel deep learning based intrusion detection system : Software defined network," Proc. 2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), IEEE, Nov. 2021, pp. 506–511, doi:10.1109/3ICT53449.2021.9581404.

[12] H. Wang, W. Li, "DDoSTC: A transformer-based network attack detection hybrid mechanism in SDN,". Sensors, vol. 21(15), Jul. 2021, doi:10.3390/s21155047.

[13] P. Choobdar, M. Naderan, M. Naderan, "Detection and multi-class classification of intrusion in Software Defined Networks using stacked auto-encoders and CICIDS2017 dataset," Wireless Personal Communications, vol. 123(1), Mar. 2022, pp. 437–471, doi:10.1007/s11277-021-09139-y

[14] T.A. Tang, L. Mhamdi, D. McLernon, S.A.R. Zaidi, M. Ghogho, "Deep Recurrent Neural Network for intrusion detection in SDN-based networks," Proc. 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), Sep. 2018, pp. 202–206, doi:10.1109/NETSOFT.2018.8460090.

[15] A. Abubakar, B. Pranggono, "Machine learning based intrusion detection system for Software Defined Networks," Proc. 2017 Seventh International Conference on Emerging Security Technologies (EST), Nov. 2017, pp. 138–143, doi:10.1109/EST.2017.8090413.

[16] K.K.L. Wong, G. Fortino, D. Abbott, "Deep learning-based cardiovascular image diagnosis: A promising challenge," Future Generation Computer Systems, vol. 110, Sep. 2020, pp. 802–811, doi:10.1016/j.future.2019.09.047.

[17] A.A. Diro, N. Chilamkurti, "Distributed attack detection scheme using Deep Learning approach for Internet of Things," Future Generation Computer Systems, vol. 82, May. 2018, pp. 761–768, doi:10.1016/j.future.2017.08.043.

[18] M.S. Elsayed, N.-A. Le-Khac, A.D. Jurcut, "InSDN: A novel sdn intrusion dataset," IEEE Access, vol. 8, Sep. 2020, pp. 165263–165284, doi:10.1109/ACCESS.2020.3022633.

[19] I. Sharafaldin, A.H. Lashkari, A.A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," Proc. 4th International Conference on Information Systems Security and Privacy (ICISSP), Jan. 2018, pp. 108-116, doi: 10.5220/0006639801080116.

[20] R. Panigrahi, S. Borah, "A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems," International Journal of Engineering & Technology, vol. 7(3.24), 2018, pp. 479–482, doi: 10.14419/ijet.v7i3.24.22797.

[21] I. Sharafaldin, A.H. Lashkari, S. Hakak, A.A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," Proc. 2019 International Carnahan Conference on Security Technology (ICCST), Oct. 2019, pp. 1–8, doi:10.1109/CCST.2019.8888419.

[22] D. Powers, "Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation," Journal of Machine Learning Technologies, vol. 2(1), Dec. 2011, pp. 37–63, doi: 10.48550/arXiv.2010.16061.