2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence) | 978-1-6654-1451-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/Confluence51648.2021.9376879

DLSDN: Deep Learning for DDOS attack detection in Software Defined Networking

Nisha Ahuja
CSE Dept.
Bennett University
Greater Noida, India
ahujanisha077@gmail.com

Gaurav Singal
CSE Dept.
Bennett University
Greater Noida, India
gaurav.singal@bennett.edu.in

Debajyoti Mukhopadhyay
CSE Dept.
Bennett University
Greater Noida, India
debajyoti.mukhopadhyay@gmail.com

***Abstract*—Software defined networking is going to be an essential part of networking domain which moves the traditional networking domain to automation network. Data security is going to be an important factor in this new networking architecture. Paper aim to classify the traffic into normal and malicious classes based on features given in dataset by using various deep learning techniques. The classification of traffic into one of the classes after pre-processing of the dataset is done. We got accuracy score of 99.75% by applying Stacked Auto-Encoder Multi-layer Perceptron (SAE-MLP) which is explained in the paper. Thus, the purpose of network traffic classification using deep learning techniques was fulfilled.**

***Index Terms*—Software-Defined Networking (SDN), Mininet Emulator, DDOS attack dataset, Deep Learning**

I. INTRODUCTION

The purpose of SDN is to centralize the network architecture. Thus in opposite of traditional network, where the two layers of control and data are tied together, in SDN there is only one control plane (SDN controller) for all the switches. Control plane perform routing of traffic and data plane is responsible for simple forwarding of the data packets.

There are nine different attacks possible on SDN namely Network Manipulation, Traffic Diversion, Side channel attack, App manipulation, Denial of Service (DoS), ARP Spoofing, API exploitation, Traffic sniffing and password guessing. The attack we are dealing with here is Distributed Denial-of-Service (DDoS) [1]. DDoS is an extended form of DoS attack. DoS is cyber-attack in which the motto is to make a machine unavailable for users by disrupting services of the host and it is accomplished by flooding the target with requests and overloading the system. In Distributed DoS (DDoS) the requests are sent from many different sources which makes stopping the attack impossible opposite to when one source is attacking because one source can be blocked but many sources are difficult to identify and block.

In this work, we have been provided with a customized Software Defined Network (SDN) based dataset provided by leadingindia.ai. In this dataset there are three different types of attacks which has been done. The various attacks which has been done are as follows:

Authors are motivated to work on this particular attack because this is the devastating attack and still work on this attack is continuing. So authors make an effort to apply the Deep learning techniques against DDOS attack. Machine learning techniques has already been applied by various authors but Deep learning techniques are more efficient and automatically detect the features which can be used to detect the attack. The different attack classes which are present in the dataset are

- Spoofed TCP-SYN attack.
- Spoofed UDP flooding attack.
- Spoofed ICMP flooding attack.

The organization of the paper is as follows: Section II discusses Related Work whereas Section III illustrates the algorithm used, Section IV illustrates the Methodology Used, Section V presents the Results and Section VI ends with Conclusion.

II. RELATED WORK

Traffic classification is one of the most important areas of network management. There are broadly three approaches for network traffic classification: port-based approach that is simple and fast but can be easily manipulated and thus not reliable; Deep packet inspection with good results but can only be used for unencrypted traffic and in real world most of data/traffic is encrypted and finally artificial-intelligence based approach. Later is considered to be reliable and is the main focus of our work for the rest of this paper. The choice of the deep neural network (DNN) based model depends highly on the dataset at hand. For an unencrypted network, traffic features can be extracted automatically using a Convolutional Neural Network (CNN).

Wang et al. [2] proposed HAST-IDS, that automatically learns network traffic features using a CNN for learning the spatial features, followed by LSTM layers to learn temporal features. So, the paper used the combination of CNN and LSTM to achieve good results. Because the features are learned automatically, it has reduced the False Alarm Rate. The author has used Darpa and ISCX-12 dataset to validate

Vinaykumar et. al. [3] proposed yet another method for using CNN and LSTM layers for network intrusion detection. Their work models network traffic as a time-series, in a predefined time range. Although their work was based on NSL-KDD dataset, we found that our dataset , after the required pre-processing becomes similar to that dataset and

Authorized licensed use limited to: Universitas Indonesia. Downloaded on January 22,2022 at 10:08:41 UTC from IEEE Xplore. Restrictions apply.

hence a similar procedure can be applied. The method is very similar to sentiment classification or time series classification with CNN-LSTM. Although the model complexity is high, and training is very slow we have included this model in our study for theoretical purposes.

Niyaz et. al. [4] advocated the use of stacked autoencoder (SAE) for dimensionality reduction for a dataset similar to ours with statistical and count features. The reduced features can then be fed to an MLP or some machine learning models of low complexity for final classification.

Phang et. al. [5] gives a novel method for DDoS attack identification along with mitigation. The method first identifies different protocols in the traffic. For each protocol data, a unique Linear support vector classifier (SVC) first classify flow entries from OpenFlow switches. In the case that a flow's position is located between two margin lines in the Linear SVC representation, inference from a Self-organizing map (SOM) is used to make a final decision.

Abdul et. al. [6] gives a SDN architecture which is a cloud based architecture. The architecture has large number of controllers, multiple switches and a monitoring cloud server which increase the reliability. The architecture includes three phases

- User checking phase.
- Controller Selection Phase.
- User and Controller interaction phase.

In this model, the traffic flows from user is analyzed by considering the following features and policies i.e. Alert, Quarantine, Block, Discard and Move. The algorithm can detect and prevent flow table attack, control plane attack and Byzantine attacks.

III. Background

In this section we describe the deep learning algorithms used for detecting the DDOS attack.

- Convolution neural network: In neural network CNN [7] is the famous model for image classification. But a variant of it, CNN-1D is applicable for text data as it is successfully applied to recommender system and various natural language processing tasks in which it give good performance. The important advantage of this algorithm is its ability to automatically detect the important features without any human intervention. The block diagram of CNN-1D is shown in figure 1 below which shows the architecture of CNN when 1-D data is presented. Figure 1 shows the architecture of Convolution Neural Network when applied to one dimensional data. When it is applied to one dimensional data the variant of Convolution 2D is used, other layers remain the same.
- Recurrent Neural Network is best for time series data [8] where current state depends on previous states. As the name suggests the network is recurrent and there is a single layer which works as many layers as shown below :
  - X (t) means input at time t

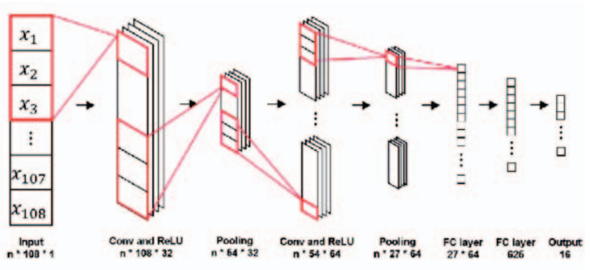

Fig. 1. CNN-1D Architecture [2]

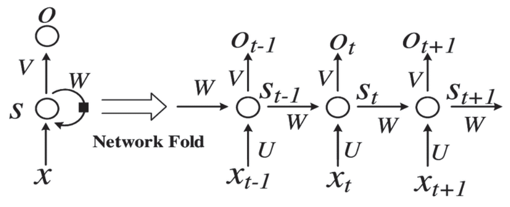

Fig. 2. RNN Architecture [9]

  - O (t) means output at time t
  - S(t) means state at time t
  - W means Weight (Constant)

  Figure 2 represents the RNN representation where the output at any time is a function of present npt and previous output. As the network gets deeper for RNN vanishing gradient problem (gradient¡1) arises where the model learns negligible, thus model isn't able to reach its optimal state or exploding gradient problem (gradient¿1) arises which is vice versa, thus model is going far from its optimal state.
- Long-Short-Term-Memory (LSTM): LSTM [10] is a special case of RNN. RNN [11] are used when the gap between the past information and place where the information is needed is small. But as the gap increases, RNN cannot work. But LSTM do not have any such disadvantage. LSTM are used to solve the problem of long term dependency. The problem of RNN has been solved by the LSTM architecture. LSTM architecture has been discussed as a combination of input gate layer and tanh layer which decides what information we are going to get from the cell state and what new information we can add to the cell state respectively. Figure 3 represents the LSTM cell which overcome the RNN shortcoming.
- CNN-LSTM: LSTM are the special case of RNN for solving long range dependency. LSTM is used for long range sequence prediction and CNN is used for feature extraction. This combination makes them suitable for various task including natural language processing problems where CNNs are used as feature extractors and LSTMs on audio and textual input data. Figure 4 shows the architecture of CNN-LSTM [13] used in our work.

Authorized licensed use limited to: Universitas Indonesia. Downloaded on January 22,2022 at 10:08:41 UTC from IEEE Xplore. Restrictions apply.

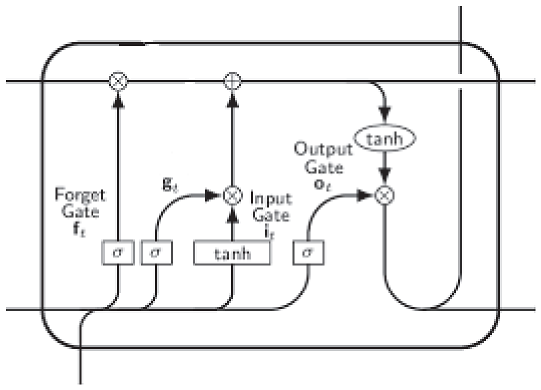

Fig. 3. LSTM Architecture [12]

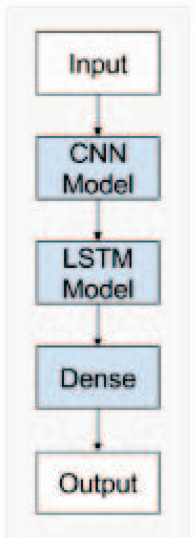

Fig. 4. CNN-LSTM Architecture

- SVC-SOM: Most of the supervised deep learning algorithms tend to have high model complexity, this results in larger time for convergence of the models. A faster and efficient method will aid in the intrusion detection process. In the method followed we use Linear SVCs along with an unsupervised deep learning method called Self Organizing Map [7](SOM) .There is a unique SVC for each protocol type present in the data. As the dataset consists of three protocols TCP, UDP and ICMP, there are three linear SVCs in the model.
  Figure 12 shows the SOM plotted for the TCP data of our dataset. The color bar shows the mean inter neuron distance where the higher value indicate a homogenous cluster and lower values indicate the less homogenous cluster. At first the Linear SVCs classify the flow entries from OpenFlow switches. The Linear Support Vector Classifier learns a linear decision boundary accompanied by two margin lines equidistant from the decision boundary, to classify points from two classes. However, depending upon the separating hyperplane and the width of the margin learned by the SVC, some data points may lie within the two margins. These data points can be misclassified as the region is allowed to encompass points from both the classes. So for the data points lying within the two margin, called suspicious points, inference from SOM is used for the final classification. SOM is an unsupervised deep learning technique. It is similar to clustering algorithms popular in data mining literature, although the method employed is very different. The clusters learned by the SOM also depend upon the neighboring clusters(neurons) in the map . This neighborhood is determined by the initial radius set for the SOM ,which decays over time. The training process should be stopped when changes made in the map start becoming minimal or insignificant. There are deeper details relating to the various hyperparameters employed. In our method we have used multiple SOMs along with multiple SVCs for the classification. This resulted in better results at an expense of greater time for learning. When we tried with three different SOMs along with three different SVCs for each protocol, we found that the later method yields more accurate results at the cost of a greater training time.

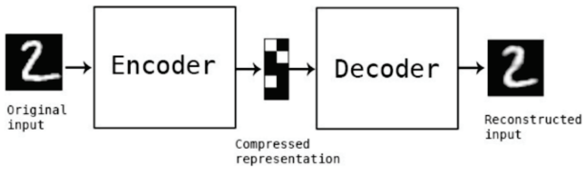

Fig. 5. SAE [14]

- SAE-MLP: Stacked autoencoder comprised of number of autoencoders tied together where the output of one autoencoder is fed to the input of other and a SoftMax classifier is later used which is used for feature extraction. Figure 5 shows the autoencoder [15] in working stage where it first encode the input and later use decoder to decode which is a compressed representation of input. An autoencoder can be think of as a neural network but the difference between the neural network and autoencoder is that autoencoder comprised of two parts Encoder and Decoder resulting in compressed representation of data than the original one. An autoencoder is an autoencoder where there is a term of sparsity penalty associated with autoencoder during training. Sparse autoencoder generally result in better learning of the model as compared to autoencoder. Sparse Autoencoder [9] is used in hybrid with MLP because Multi Layer Perceptron has the curse of dimensionality and SAE is good for dimensionality reduction. The only thing that is different is we used two different optimizers, sgd for the first 10 epochs and

Authorized licensed use limited to: Universitas Indonesia. Downloaded on January 22,2022 at 10:08:41 UTC from IEEE Xplore. Restrictions apply.

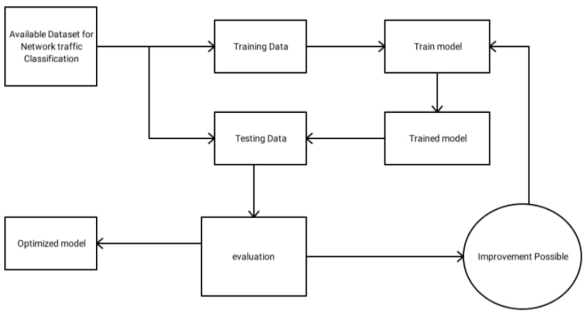

Fig. 6. Block diagram of methodology followed.

Adam for next 150 epochs, this reduced the training time considerably. SGD worked with a higher learning rate while Adam worked with default initial learning rate. Thus, with a hybrid of SAE-MLP [16] the curse of dimensionality from MLP was lifted and it performed considerably better.

IV. METHOOLOGY

In this section, the method which has been followed in the paper is discussed. Figure 6 below shows the methodology which has been followed to achieve the results. We also show the step by step procedure of methodology.

1) Data Pre-Processing: The dataset is given to us by leading India Project Mentor and consists of 22 features and available in the Mendeley data repository given in footer [1] and are mostly statistical features. In our dataset, there are three different protocols in the traffic flows TCP, UDP and ICMP. The label denotes whether the traffic is attack or normal. The dataset consists of features defining the source and destination along with providing information about the bytes, packets, duration of transfer, speed of transfer etc. Pre-processing involves removing the redundant entries in the dataset, the redundant entries were the ones with packet rate or byte per flow or packet per flow as zero because no data exchange is taking place and thus those entries were dropped. Next step was identifying the type of the variables i.e. whether the variable is categorical or numerical and then encoding the categorical variables. Here, categorical variables were source, destination, switch and protocol. All the categorical variables [12] were encoded by using one- hot-encoding. Then we tried to find the correlation of the input features with the output features and the accuracy of various machine learning classifiers and deep neural networks by heatmap plot and correlation techniques. The data column was found to be redundant while all other features held importance, thus the date feature was dropped. Lastly normalization [17] of the numeric features was performed and the pre-processing was done.
2) Apply Machine Learning classifiers: In this step we apply various Deep learning classifiers which have been discussed in section 3. The dataset is trained on a particular model and then it is tested on the unseen data.
3) Evaluation :In this step the applied models were evaluated using metrics of accuracy, precision, recall, F-score, False positive rate and false negative rate.

[1]https://data.mendeley.com/datasets/jxpfjc64kr/1

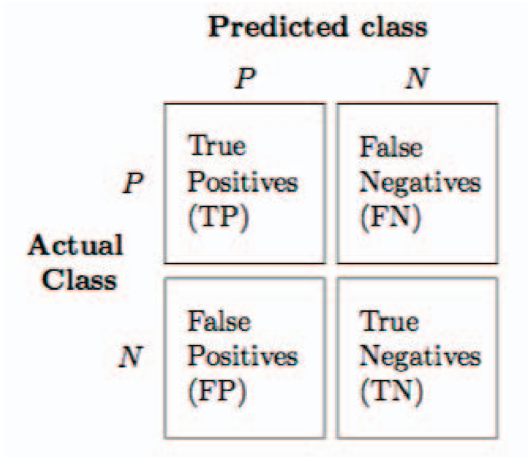

Fig. 7. Confusion Matrix [11]

- Accuracy= t_p+t_n/t_p+t_n+f_p+f_n
- Precision= t_p/t_p+f_p
- Recall= t_p/t_p+f_n
- F1-Score=2*Precision*Recall/Precision+Recall
  Here, t_p - true positives t_n - true negatives f_p– false positives f_n – false negatives

These values are derived from something known as confusion matrix shown in figure 7, which gives a comparison of the true labels to labels predicted by the model. True negative is the count of values, which are actually not positive and also predicted to be not positive. False negative is the count of values which are predicted to be not positive but actually positive. False positive is the count of values which are predicted to be positive but actually not positive.
Thus, accuracy is the number of correctly predicted values divided by total number of predictions. Positive precision is number of correctly predicted positive values from the total number of values predicted as positive. Negative precision is number of correctly predicted negative values from the total number of values predicted as negative. Positive recall is total number of correctly predicted positive values over total actual positive values. Negative recall is total number correctly predicted negative predictions over total actual negative values. F1-score is the harmonic mean [14] of precision

Authorized licensed use limited to: Universitas Indonesia. Downloaded on January 22,2022 at 10:08:41 UTC from IEEE Xplore. Restrictions apply.

TABLE I
RESULTS OF DIFFERENT MODELS

Model	*Acc*	NPrec	*APrec*	NRecall	*ARecall*	NFScore	*AFScore*	FPR	*FNR*
CNN	0.9874	0.9875	0.9873	0.9890	0.9855	0.9883	0.9864	0.0144	0.0109
LSTM	0.9560	0.9620	0.9490	0.9564	0.9556	0.9592	0.9523	0.0443	0.0435
CNN-LSTM	0.9948	0.9943	0.9955	0.9966	0.9926	0.9954	0.9940	0.0073	0.0033
SVC-SOM	0.9545	0.9671	0.9375	0.9540	0.9551	0.9605	0.9462	0.0448	0.0459
SAE-MLP	0.9975	0.9996	0.9969	0.9977	0.9994	0.9987	0.9982	0.0005	0.0022

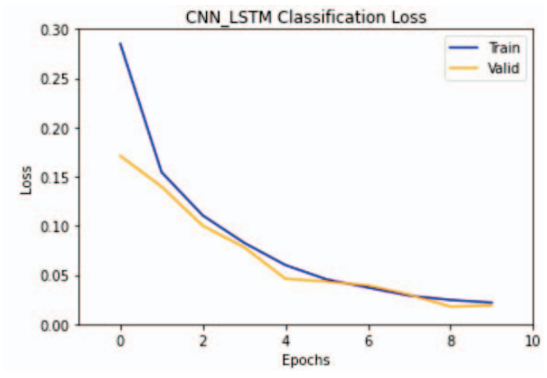

Fig. 8. CNN-LSTM Classification loss

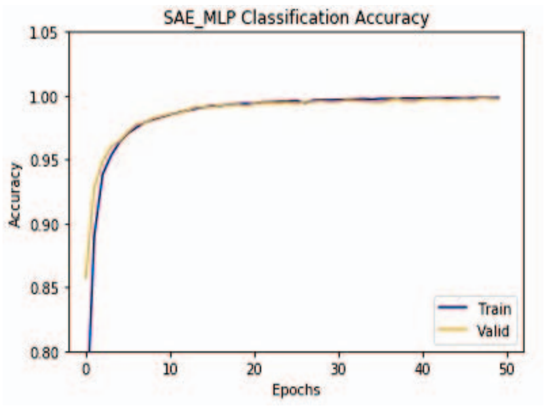

Fig. 10. SAE-MLP classification accuracy

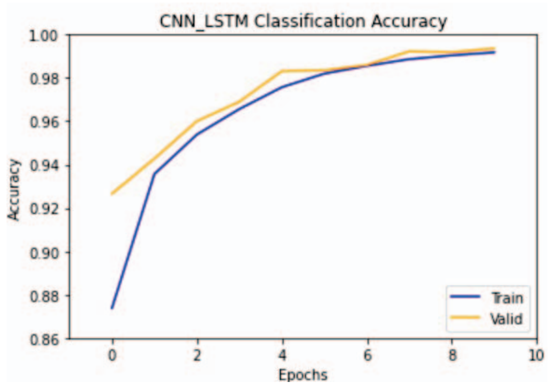

Fig. 9. CNN-LSTM classification accuracy

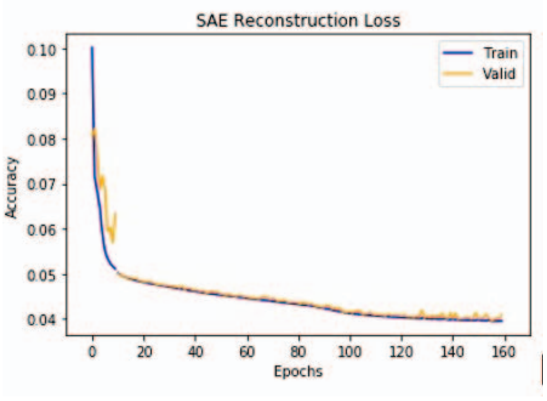

Fig. 11. SAE Reconstruction loss

and recall. False positive rate is count of actual false positive values over number of predicted positive values. False negative rate is the count of actual false negatives over number of predicted negatives. Accuracy, precision, Recall and F1-score is to be maximized while false positive rate and false negative rate is to be minimized for optimization.

4) Hyperparameter Tuning: After the evaluation of the dataset is done further improvements can be done by means of hyper-parameter tuning. Various hyperparameters which has been used included are the number of layers, the number of epoch, the regularization parameter which when tuned give optimized results.

V. RESULT

In this section we show the result obtained in terms of the accuracy achieved when applying a classification algorithm.

Figure 8, 9 shows the CNN-LSTM Classification loss and accuracy vs number of epoch. Figure 10, 11 shows the SAE Reconstruction accuracy and loss vs number of epoch. Lesser the loss higher is the reconstruction accuracy. Table 1 shows the accuracy of different classification algorithms with the highest classification accuracy of 99.75% with SAE-MLP algorithm. Figure 13 shows the classifications models accuracy comparison chart.

Authorized licensed use limited to: Universitas Indonesia. Downloaded on January 22,2022 at 10:08:41 UTC from IEEE Xplore. Restrictions apply.

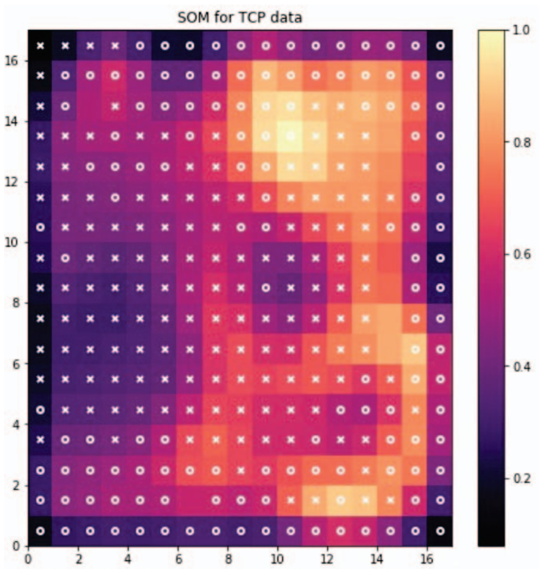

Fig. 12. SOM for TCP data

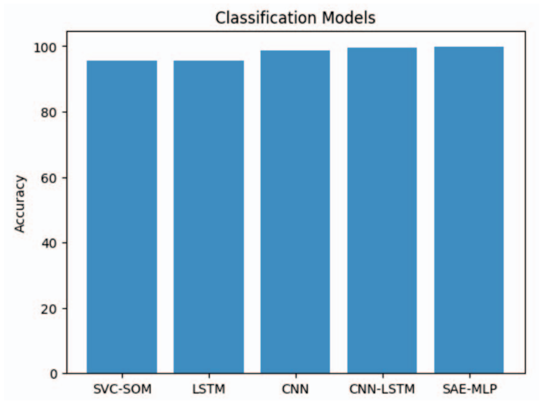

Fig. 13. Comparison of classification models

VI. Conclusion

We have done traffic classification of Software defined network traffic which was given to us by Leading India. We have used various deep learning techniques for classifying the traffic into normal or malicious classes. Out of which Stacked Auto-Encoder-MLP achieved highest accuracy of 99.75%.

References

[1] D. Mukhopadhyay, B.-J. Oh, S.-H. Shim, and Y.-C. Kim, "A study on recent approaches in handling ddos attacks," *arXiv preprint arXiv:1012.2979*, 2010.

[2] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu, "Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2017.

[3] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2017, pp. 1222–1228.

[4] Q. Niyaz, W. Sun, and A. Y. Javaid, "A deep learning based ddos detection system in software-defined networking (sdn)," *arXiv preprint arXiv:1611.07400*, 2016.

[5] T. V. Phan, N. K. Bao, and M. Park, "A novel hybrid flow-based handler with ddos attacks in software-defined networking," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE, 2016, pp. 350–357.

[6] I. Abdulqadder, D. Zou, I. Aziz, B. Yuan, and W. Dai, "Deployment of robust security scheme in sdn based 5g network over nfv enabled cloud environment," *IEEE Transactions on Emerging Topics in Computing*, 2018.

[7] H. AlMomin and A. A. Ibrahim, "Detection of distributed denial of service attacks through a combination of machine learning algorithms over software defined network environment," in *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. IEEE, 2020, pp. 1–4.

[8] A. AlEroud and I. Alsmadi, "Identifying cyber-attacks on software defined networks: An inference-based intrusion detection approach," *Journal of Network and Computer Applications*, vol. 80, pp. 152–164, 2017.

[9] A. S. da Silva, J. A. Wickboldt, L. Z. Granville, and A. Schaeffer-Filho, "Atlantic: A framework for anomaly traffic detection, classification, and mitigation in sdn," in *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2016, pp. 27–35.

[10] L. F. Maimó, Á. L. P. Gómez, F. J. G. Clemente, M. G. Pérez, and G. M. Pérez, "A self-adaptive deep learning-based system for anomaly detection in 5g networks," *IEEE Access*, vol. 6, pp. 7700–7712, 2018.

[11] Z. Al Haddad, M. Hanoune, and A. Mamouni, "A collaborative network intrusion detection system (c-nids) in cloud computing," *International Journal of Communication Networks and Information Security*, vol. 8, no. 3, p. 130, 2016.

[12] M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, and S. Vasupongayya, "Advanced support vector machine-(asvm-) based detection for distributed denial of service (ddos) attack on software defined networking (sdn)," *Journal of Computer Networks and Communications*, vol. 2019, 2019.

[13] S. Sen, K. D. Gupta, and M. M. Ahsan, "Leveraging machine learning approach to setup software-defined network (sdn) controller rules during ddos attack," in *Proceedings of International Joint Conference on Computational Intelligence*. Springer, 2020, pp. 49–60.

[14] C. Buragohain and N. Medhi, "Flowtrapp: An sdn based architecture for ddos attack detection and mitigation in data centers," in *2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 2016, pp. 519–524.

[15] A. Panda, S. S. Samal, A. K. Turuk, A. Panda, and V. C. Venkatesh, "Dynamic hard timeout based flow table management in openflow enabled sdn," in *2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*. IEEE, 2019, pp. 1–6.

[16] N. Dayal and S. Srivastava, "An rbf-pso based approach for early detection of ddos attacks in sdn," in *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*. IEEE, 2018, pp. 17–24.

[17] J. Manan, A. Ahmed, I. Ullah, L. Merghem-Boulahia, and D. Gaïti, "Distributed intrusion detection scheme for next generation networks," *Journal of Network and Computer Applications*, vol. 147, p. 102422, 2019.

Authorized licensed use limited to: Universitas Indonesia. Downloaded on January 22,2022 at 10:08:41 UTC from IEEE Xplore. Restrictions apply.