Deep Learning-based Slow DDoS Attack Detection in SDN-based Networks

Beny Nugraha[1,2], Rathan Narasimha Murthy[1]
[1]Technische Universität Chemnitz, Chemnitz, Germany
[2]Department of Electrical Engineering, Mercu Buana University, Jakarta, Indonesia
Email: beny.nugraha@etit.tu-chemnitz.de | rathan.narasimha@gmail.com

***Abstract*—Software-Defined Networking (SDN) is a promising networking paradigm that provides outstanding manageability, scalability, controllability, and flexibility. Despite having such promising features, SDN is not intrinsically secure. For instance, it still suffers from Denial of Service (DDoS) attacks, which is one of the major threats that compromise the availability of the network. One type of DDoS attacks, that is considered as one of the most challenging to be detected, are slow DDoS attacks. In recent years, deep learning algorithms have been applied for reliable and highly accurate traffic anomaly detection. Therefore, in this paper, we propose the use of a hybrid Convolutional Neural Network-Long-Short Term Memory (CNN-LSTM) model to detect slow DDoS attacks in SDN-based networks. The performance of this method is evaluated based on custom datasets. The obtained results are quite impressive – all considered performance metrics are above 99%. Our hybrid CNN-LSTM model also outperforms other deep learning models like MultiLayer Perceptron (MLP) and standard machine learning models like 1-Class Support Vector Machines (1-Class SVM).**

Keywords—Slow DDoS Attack, Software-Defined Networking, Deep Learning, Performance Evaluation, Supervised Learning

I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks are one of the hazardous threats for Internet services. Amazon Web Services (AWS) reported that in Q1 2020 they withstand a 2.3 Tbit/s DDoS attack [1], which is the largest attack so far. DDoS attacks are still emerging, which is a severe threat. Initially, a large number of resources are required to initiate a DDoS attack, for instance, several computers or bots are needed to exhaust the resources of an application server. In recent years, the introduction of slow DDoS attacks even more increased the threat level of DDoS attacks. Slow DDoS attacks are quite different compared to conventional DDoS flooding attacks. They aim to disrupt the service provided by an application server by sending a small amount of data over a long period [2]. Comparatively, fewer resources are required to launch slow DDoS attacks compared to DDoS flooding attacks. Therefore, slow DDoS attacks are more affordable and can even be launched from a single machine with a slow Internet connection. Furthermore, slow DDoS attacks are more difficult to detect because they mimic the traffic pattern of legitimate traffic.

Contrary to conventional networks, Software-Defined Networks (SDN) might be able to provide a reliable defense mechanism against slow DDoS attacks [3]. SDN offers the following features which can be beneficially applied for this task: a global view of the whole network's state and fine-grained monitoring of the network flows [4]. However, SDN is not intrinsically secure w.r.t. slow DDoS attacks – the studies in [5] and [6] stated, that such attacks can saturate the SDN data plane slowly and silently. Fig. 1 shows a slow DDoS attack scenario in an SDN-based network. Several researchers have already proposed methods to detect slow DDoS attacks in SDN [5]-[8]. However, their approaches exhibit the following limitations: they either rely on a pre-defined threshold that has to be set manually [5], [8] or depend on a dynamic timeout of the flow rules within the SDN switches which leads to a forwarding delay increase [6] or requires a high management effort to remember connection state information [7]. To overcome these limitations, deep learning, which is a subcategory of machine learning, has recently become a hot candidate for providing early and highly accurate slow DDoS attack detection without any need to manually set a pre-defined threshold [9].

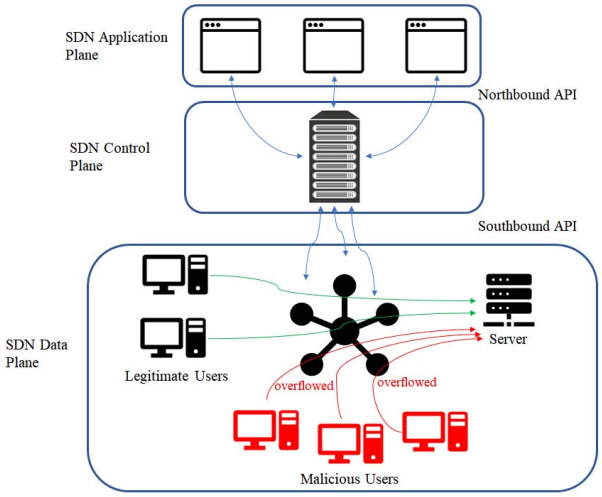

Fig. 1. Example slow DDoS attacks scenario in an SDN-based network

In this paper, we propose and implement a deep learning-based slow DDoS attack detection approach for an SDN-based network. A hybrid Convolutional Neural Network-Long-Short Term Memory (CNN-LSTM) model constitutes the core of the slow DDoS detection module. We apply a supervised learning technique for the deep learning-based classification task and several performance metrics are used to evaluate the performance of our method. The workflow is as follows: First, we create datasets for benign and slow DDoS attack traffic flows. This is required as datasets for slow DDoS attack traffic are not publicly available yet. For creating the datasets, we

978-1-7281-8159-2/20/$31.00 ©2020 IEEE

emulate an SDN-based network, synthetically generate benign and attack traffic flows and utilize the ability of the SDN switches to collect traffic flow statistics. Second, our CNN-LSTM model is trained, validated, and tested based on the created datasets. Hyperparameter tuning is performed to determine the best set of hyperparameters w.r.t. several performance metrics. Finally, we compare the performance of the hybrid CNN-LSTM model with another deep learning model, namely MultiLayer Perceptron (MLP), and a standard machine learning technique, namely 1-Class Support Vector Machine (1-Class SVM).

The remaining of this paper is organized as follows: Section II provides an overview of existing detection mechanisms for slow DDoS attacks. Section III discusses the implementation of our deep learning-based slow DDoS attack detection approach in an SDN-based network. In Section IV the results of our performance analysis are presented and discussed. Finally, Section V concludes the paper by summarizing the main findings and outlining some ideas for future work.

II. Existing Detection Mechanisms for Slow DDoS Attacks

Several researchers have proposed methods to solve the slow DDoS attack detection problem in SDN-based networks. Pascoal et al. [5] introduce a method with a pre-defined flow rule threshold: when the number of flow rules exceeds the threshold, the SDN switches will drop some flow rules randomly. This simple method could cause a problem when the flows from legitimate users are dropped. A scheme applying dynamic flow rule timeout and random flow rule creation delay for incoming flows are introduced by Cao et al. [6]. Their method could stress the SDN controller because it needs to process more packet-in messages. In addition, packets from legitimate users will suffer from additional forwarding delay. Lukaseder et al. [7] propose to monitor all the incoming flows and then determine the malicious flows by calculating a so-called suspiciousness score. Their method requires a high effort to remember every flow feature information. Similar to Pascoal et al. [5], Hong et al. [8] introduce a pre-defined threshold to limit the number of incomplete HTTP requests, and the incomplete requests are dropped if their number exceeds the threshold. The drawback of their method is that a unique pre-defined threshold cannot be applied to all servers because each server might require an individual threshold. To conclude, all the above-mentioned mechanisms for detecting slow DDoS attacks in SDN-based networks have some severe drawbacks. However, these drawbacks can be circumvented by applying machine learning or deep learning-based detection techniques. To the best of our knowledge, the first method that used machine learning-based detection for slow DDoS attacks in SDN-based networks is proposed by Phan et al. [10]. They utilized a reinforcement learning framework to provide early detection of slow DDoS attacks. They considered several machine learning techniques namely SVM with supervised learning, Random Forest with supervised learning, and Self Organizing Maps with unsupervised learning and emulated 4000 benign traffic and 4000 attack traffic flows (i.e. a total of 8000 traffic flows), which is sufficient for machine learning techniques. However, considering the continuous huge increase of Internet traffic, conventional machine learning techniques pose one major disadvantage: they heavily rely on feature engineering which usually involves human interaction and takes a lot of effort

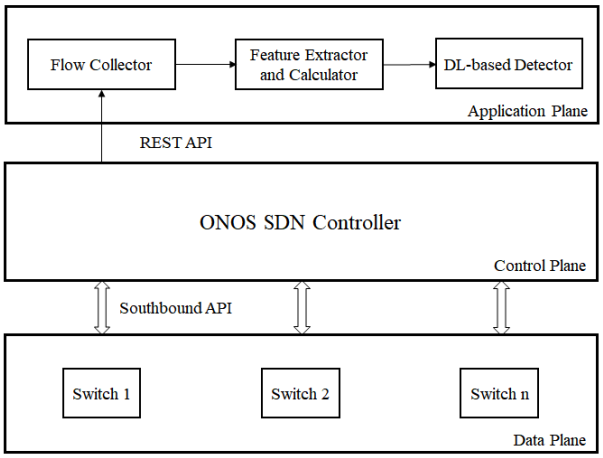

Fig. 2. The architecture of the DL-based slow DDoS attack detection framework for SDN-based networks

[11]. With deep learning techniques, the extraction of complex features is performed in an autonomous manner which results in higher accuracy and faster processing compared to conventional machine learning techniques [12]. Therefore, deep learning techniques are well-suited to handle large datasets. This inspired us to introduce deep learning-based slow DDoS attack detection in SDN-based networks. To the best of our knowledge, this approach was not yet considered by other researchers for slow DDoS attack detection.

III. Deep Learning-based Slow DDoS Attack Detection in SDN-based Networks

In this section, the implementation of our deep learning-based slow DDoS attack detection mechanism in an SDN-based network is described.

A. DDoS Attack Detection Architecture and Operational Workflow

The architecture of our slow DDoS attack detection framework is depicted in Fig. 2. It consists of the following modules: flow collector, feature extractor and calculator, and deep learning-based detector. These modules are implemented as SDN applications using the Python programming language.

1) Flow Collector: In contrast to conventional IP networks, the traffic forwarding in an SDN-based network is performed in a per-flow granularity and traffic flow information is available from the SDN switches. The flow collector module is responsible for requesting the flow statistics periodically through the REST API from the SDN controller. It also checks the source IP address of the host, which is trying to establish a connection to the server. The collection process works as follows: for each source IP address the respective flows are monitored during a time interval of three seconds and the flow statistics are then aggregated into one entry which is forwarded to the next module. Thus, one entry comprises one source IP address and the statistics of one or multiple flows.

2) Feature Extractor and Calculator: This module, after having received an entry from the flow collector module, extracts the raw features and calculates additional features which are relevant for slow DDoS attack detection (for more details see chapter III. C).

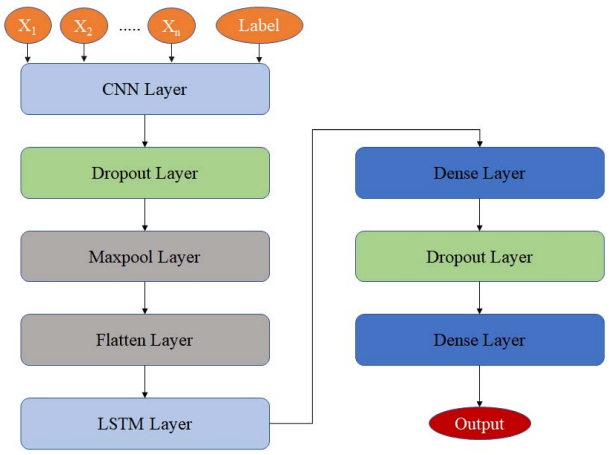

Fig. 3. Hybrid CNN-LSTM model

3) Deep learning-based Detector: The hybrid CNN-LSTM model is used for the detection of slow DDoS attack traffic. This model is chosen as it has been proven by [13] to yield high Accuracy and Recall when detecting a variety of DDoS attacks. The detector module analyzes the output from the feature extractor module and decides whether the entries are classified as attack or benign traffic. It therefore acts as a binary classifier assigning the label '1' for slow DDoS attack traffic and the label '0' for benign traffic. The labels are related to the source IP address because one host can be either a legitimate or malicious one. Fig. 3 depicts the hybrid CNN-LSTM model which is used within the detector module. The indices $x_1 - x_n$ denote the features associated with an entry, i.e. each entry is fed into the model as {feature 1, feature 2, … , feature n, and label}.

The first CNN layer is a 1D-CNN layer. It is chosen instead of a 2D-CNN layer because its training time is faster and it is not required to run on a dedicated Graphical Processing Unit (GPU) [14]. Unlike the hybrid CNN-LSTM model from [13], we include three additional layers between the CNN and the LSTM layer: Dropout, Maxpool, and Flatten layer. The Dropout layer is used in order to prevent overfitting, the Maxpool layer extracts the most relevant features [15] and the Flatten layer helps in accelerating the learning process [16]. The subsequent LSTM layer has two roles: first, it learns the characteristics of the current entry, and second, it stores information from previous entries [17]. A fully connected dense layer with Rectified Linear Unit (ReLU) activation function is placed after the LSTM layer. The study in [18] showed that ReLU achieves a faster learning time and better results (w.r.t. several performance metrics) than other activation functions e.g. the tanh function. Another dropout layer is provided subsequently, and the last layer is a dense layer with sigmoid activation function that is used to finally perform the binary classification into slow DDoS attack and benign traffic entries [19].

B. Feature Extraction

As mentioned, each entry that is received by the feature extractor and calculator module can contain information about one or multiple flows. The following six features can be extracted directly from each entry: source IP address, protocol type, destination port, total number of flows, total number of packets, and total number of bytes. In addition, we propose to consider the following six features (where the first four features were also recommended in [10]) that might be relevant to differentiate between slow DDoS attack and benign traffic entries (thus we consider 12 features in total):

1) Average number of packets per flow (APf): In a slow DDoS attack, instead of sending a large number of packets at once, the attacker sends a very small number of packets at a very low rate. For instance, an attacker might send 5 packets per flow, whereas a legitimate user sends more than 300 packets per flow.

2) Average packet size per flow (ABf): In a slow DDoS attack, the packet size is smaller compared to benign traffic. The attacker's goal is to generate packets that meet the minimum size requirement [20]. For instance, an attacker might generate traffic with around 64-70 byte long packets whereas the packet size in benign traffic is usually between 64-1518 bytes [20].

3) Packet change rate (PCR): A slow DDoS attacker tries to keep the maximum number of connections alive. The attacker can achieve this by changing the number of packets sent to the server. The packet change rate denotes the change of the number of packets sent in subsequent three second intervals.

4) Flow change rate (FCR): In a slow DDS attack, an attacker increases the number of flows slowly in order to cause a flow table overflow in the SDN switch. Once this happens, the traffic flow entries stemming from legitimate users will be discarded after an idle-timeout period of 10 seconds [6]. The flow change rate denotes the change of the number of flows sent in subsequent three second intervals.

5) Average number of packets per second (APs): This feature represents the average number of packets that are observed per second. A small value could indicate slow DDoS attack traffic.

6) Average number of bytes per second (ABs): This feature indicates the average number of bytes per second. A smaller than usual *ABs* value could indicate a slow DDoS attack.

C. Performance Metrics

The performance of our detection system is evaluated by using the metrics Accuracy, Precision, Specificity, Recall, and F1 score. These metrics are calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (1)$$

$$Precision = \frac{TP}{(TP + FP)} \quad (2)$$

$$Specificity = \frac{TN}{(TN + FP)} \quad (3)$$

$$Recall = \frac{TP}{(TP + FN)} \quad (4)$$

$$F1\ Score = 2\left(\frac{Recall \times Precision}{Recall + Precision}\right) \quad (5)$$

True Positive (TP) is the number of correctly predicted slow DDoS attack traffic entries. True Negative (TN) represents the number of correctly predicted benign traffic entries. False Positive (FP) reflects the number of benign traffic entries that are predicted as slow DDoS attack traffic entries, and False Negative (FN) represents the number of slow DDoS attack traffic entries that are predicted as benign traffic entries. Accuracy in classification problems denotes the ratio of correct predictions and all predictions made. Precision represents the ratio of correctly identified slow DDoS attack traffic entries and the total number of predicted slow DDoS attack traffic entries. Specificity is the ratio of correctly identified benign traffic entries and the total number of benign traffic entries that are actually happening. Recall is the ratio of correctly identified slow DDoS attack traffic entries and the total number of slow DDoS attack traffic entries that are actually appearing. Lastly, F1 Score is an overall performance measure balancing Precision and Recall by calculating the harmonic mean of both values.

IV. Performance Analysis

A. Benign and Slow DDoS Attack Traffic Datasets

The evaluation of our detection framework is performed by using offline datasets. We emulate a small SDN network topology and synthetically generate both benign and slow DDoS attack traffic flows as suitable datasets were not publicly available at the time of this work. Our network scenario comprises two SDN switches, a Web server as the victim, 6 legitimate hosts, and 4 malicious hosts, see Fig. 4. The flow collector module monitors the flow statistics of Switch 2, which is connected to the victim. The benign traffic flows are generated using the Hping3 traffic flow generator while the slow DDoS attack traffic flows are generated using several open-source slow DDoS attack tools [21]-[25]. The benign traffic flows are UDP and HTTP flows while the slow DDoS attack traffic flows are HTTP flows. In total, we generate 468,002,403 traffic flows consisting of 355,758,086 benign and 112,244,317 slow DDoS attack flows. By using the flow collector module, these flows are merged into 1.000.000 entries (600.000 benign traffic entries and 400.000 slow DDoS attack traffic entries). The dataset is then split into 80% training and validation and 20% testing data, respectively. The training and validation dataset in turn is split into 80% training data and 20% validation data.

It should be noted that in our datasets the values ranges of the features are quite different. Therefore, it is recommended to scale these values in order to prevent larger values from dominating the smaller ones in the classification process [26]. As the features in our dataset contain real-valued numbers, we use the Min-Max Scaler [26] which linearly normalizes all feature values to lie in the range (0,1). The Min-Max Scaler is also used in previous works regarding deep learning-based intrusion detection [27][28].

B. Hyperparameter Tuning

Hyperparameter tuning is a procedure to determine the best set of hyperparameters w.r.t. the considered performance metrics. In this work, we tune the following hyperparameters by performing several experiments: learning rate, dropout rate, and CNN filter and kernel size. For all experiments, we set the number of epochs to 50.

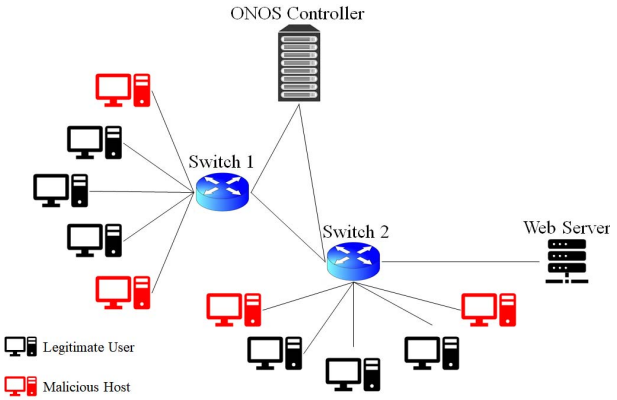

Fig. 4. SDN Network Scenario

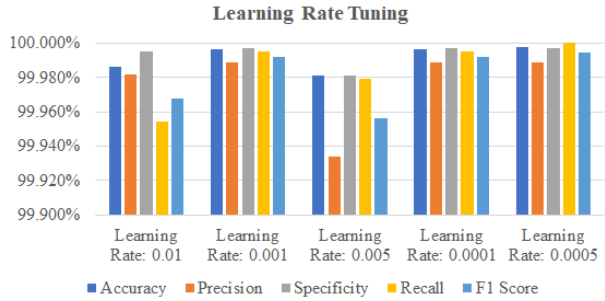

Fig. 5. Learning rate tuning

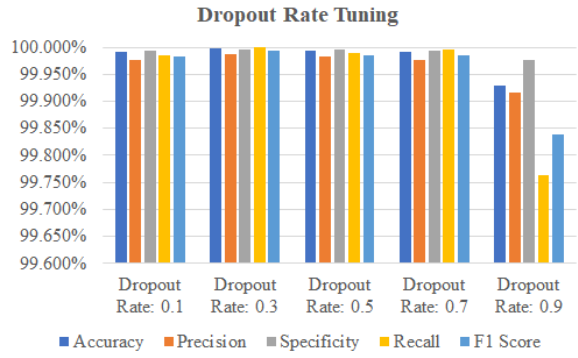

Fig. 6. Dropout rate tuning

1) Learning Rate

The learning rate is the most common hyperparameter to be optimized as it is directly related to the learning speed at which the model can enhance its detection accuracy [29]. In this work, we determine the optimum learning rate out of the set {0.01, 0.001, 0.005, 0.0001, 0.0005}, similar to the learning rate tuning applied in [30]. The result of the learning rate tuning is depicted in Fig. 5. It turns out that the optimum learning rate is 0.0005.

2) Dropout Rate

The Dropout rate is the probability with which randomly selected neurons will be dropped or ignored during training. For instance, a dropout rate of 0.5 means that 50% of the neurons will be dropped [31]. Dropout rate optimization is also mentioned in [32], however, the authors did not state the dropout rate value which they finally used. In this work, we determine the optimum dropout rate value out of the set {0.1, 0.3, 0.5, 0.7, 0.9}. We keep the optimum learning rate (0.0005) that has been determined beforehand. The results of the dropout rate tuning can be seen in Fig. 6. It turns out that

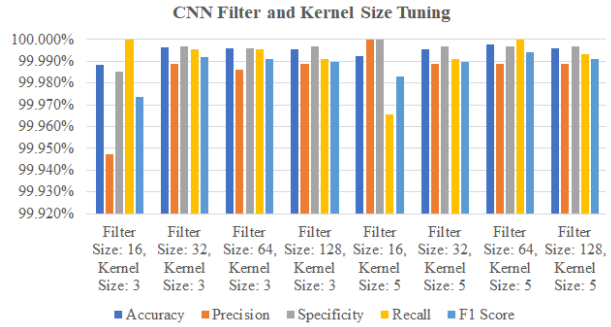

Fig. 7. CNN filter and kernel size tuning

TABLE I. Optimum set of hyperparameters

Hyperparameter	Value
Learning Rate	0.0005
Dropout Rate	0.3
CNN Filter and Kernel Size	64 / 5

a dropout rate of 0.3 is optimum, even though there are only small differences in the performance obtained for different dropout rates.

3) CNN Filter and Kernel Size

An optimum value for the CNN filter and kernel size is mentioned in [15], [33], and [34]. However, the respective tuning process is not explained. In this work, we determine the optimum CNN filter and kernel size out of the following options: {Filter Size: 16 & Kernel Size: 3, Filter Size: 32 & Kernel Size: 3, Filter Size: 64 & Kernel Size: 3, Filter Size: 128 & Kernel Size: 3, Filter Size: 16 & Kernel Size: 5, Filter Size: 32 & Kernel Size: 5, Filter Size: 64 & Kernel Size: 5, Filter Size: 128 & Kernel Size: 5}. We apply the optimum learning and dropout rates that have been determined beforehand. Fig. 7 depicts the results of the CNN filter and kernel size tuning. It turns out that the combination Filter Size: 64 and Kernel Size: 5 yields the best result.

A summary of the results of our hyperparameter tuning process is shown in Table I. By using these hyperparameters, our slow DDoS attack detection model achieves an Accuracy of 99.998%, a Precision of 99.989%, a Specificity of 99.997%, a Recall of 100%, and an F1 Score of 99.994%.

C. Performance Comparison with MultiLayer Perceptron (MLP) and 1-Class SVM

We compare the performance of our hybrid CNN-LSTM model with another deep learning model and a conventional machine learning technique, namely the MLP and the 1-Class SVM, respectively. The MLP model is chosen because it is regarded as a good model for outlier (anomaly) detection and has recently been identified in [35] as a highly accurate detection mechanism. The values of the learning and dropout rates in the MLP model are chosen identically to the values we have determined for our hybrid CNN-LSTM model beforehand (0.0005 and 0.3, respectively). The MLP model consists of an input layer with ReLU activation function followed by a dropout layer and followed by two dense layers and another dropout layer. The output from this dropout layer is fed into a dense layer and then to a fully connected layer. The last layer is a dense layer with a sigmoid activation function.

For comparing the performance of the 1-Class SVM to that of our model, we use the four features that are regarded as the optimal features in [10], namely: average number of packets per flow, average packet size per flow, packet change ratio, and flow change ratio. We investigate the performance of the 1-Class SVM using 120.000 benign traffic entries and 80.000 slow DDoS attack traffic entries that are taken from our dataset. It should be noted that we apply the 1-Class SVM on a dataset that is larger than the original dataset in [10].

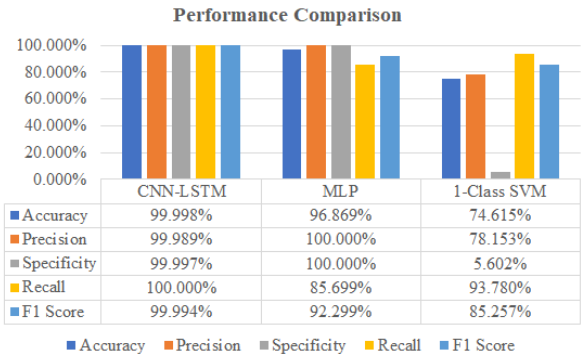

Fig. 8. Performance comparison

The result of the performance comparison is presented in Fig. 8. It can be seen that the MLP model can achieve 100% Precision and Specificity, but it can only correctly detect 85.7% of the slow DDoS traffic entries. Meanwhile, the 1-Class SVM is able to detect 93.8% of the slow DDoS traffic entries, however, its other performance metrics are significantly lower than for the MLP and the hybrid CNN-LSTM model. The hybrid CNN-LSTM model achieves more than 99% in all performance metrics. Therefore, it can be concluded that our hybrid CNN-LSTM model yields the overall best performance.

We also compared the execution time of the MLP model and the hybrid CNN-LSTM model required for performing the training. We observed that the MLP model requires less time than the hybrid CNN-LSTM model, the reason might be that the hybrid CNN-LSTM model requires a 3D data format as input, so that the dataset needs to be converted accordingly [13].

V. Conclusions

In this paper, we present a deep learning-based framework to detect slow DDoS attacks. The detection module collects traffic flow statistics from SDN switches via the SDN controller's REST API and analyzes it to detect a slow DDoS attack. We implemented the hybrid CNN-LSTM model in the detection module. After determining the best set of hyperparameter values, we carried out several experiments using a synthetically generated traffic flow dataset to evaluate the detection performance of our hybrid CNN-LSTM model. The obtained results show that this model performs exceptionally well (i.e., it achieves more than 99% in all considered performance metrics). Lastly, we compared the performance of the hybrid CNN-LSTM model with that of the MLP and the 1-Class SVM models. It turned out that the hybrid CNN-LSTM model outperforms both the MLP and the 1-Class SVM model. It also can be stated that when dealing with a large dataset, the deep learning models (hybrid CNN-LSTM and MLP) perform better than the machine learning model (1-Class SVM).

In future, we intend to apply continual learning to address the forgetting issue that might occur in a neural network. By

that, the detection framework will be able to detect traffic anomalies in real-time. Furthermore, we want to investigate novel attack mitigation techniques that allow to terminate a slow DDoS attack closer to the sources of the attack.

ACKNOWLEDGMENT

This work has been performed in the framework of the Celtic-Plus project SENDATE Secure-DCI, funded by the German BMBF (ID 16KIS0481).

REFERENCES

[1] Amazon Web Service, "Threat Landscape Report – Q1 2020", AWS Shield, 2020.

[2] E. Cambiaso and et al., "Slow dos attacks: definition and categorisation," *International Journal of Trust Management in Computing and Communications*, vol. 1, no. 3-4, pp. 300–319, 2013.

[3] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 602–622, 2016.

[4] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. "Software-Defined Networking: A Comprehensive Survey," in *Proceedings of the IEEE 103.1 (Jan. 2015)*, pp. 14–76. doi: 10.1109/JPROC.2014.2371999.

[5] T. A. Pascoal, Y. G. Dantas, I. E. Fonseca, and V. Nigam, "Slow tcam exhaustion ddos attack," in *IFIP International Conference on ICT Systems Security and Privacy Protection*, pp. 17–31, Springer, 2017.

[6] J. Cao, M. Xu, Q. Li, K. Sun, Y. Yang, and J. Zheng, "Disrupting sdn via the data plane: a low-rate flow table overflow attack," in *International Conference on Security and Privacy in Communication Systems*, pp. 356–376, Springer, 2017.

[7] T. Lukaseder, L. Maile, B. Erb, and F. Kargl, "Sdn-assisted network-based mitigation of slow ddos attacks," in *CoRR*, vol. abs/1804.06750, 2018.

[8] K. Hong, Y. Kim, H. Choi, and J. Park, "Sdn-assisted slow http ddos attack defense method," in *IEEE Communications Letters*, vol. 22, pp. 688–691, April 2018.

[9] D. Berman, A. Buczak, J. Chavis and C. Corbett, "A Survey of Deep Learning Methods for Cyber Security," in *Information*, vol. 10, no. 4, p. 122, 2019.

[10] T. Phan, T. Gias, S. Islam, T. Huong, N. Thanh and T. Bauschert, "Q-MIND: Defeating Stealthy DoS Attacks in SDN with a Machine-learning based Defense Framework," in *IEEE Global Communications Conference (IEEE GLOBECOM)*, Waikoloa, HI, USA, 2019.

[11] M. Najafabadi, F. Villanustre, T. Khoshgoftaar, N. Seliya, R. Wald and E. Muharemagic, "Deep learning applications and challenges in big data analytics," in *Journal of Big Data*, vol. 2, no. 1, 2015.

[12] J. Schmidhuber, "Deep learning in neural networks: An overview," in *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.

[13] M. Roopak, G. Yun Tian and J. Chambers, "Deep Learning Models for Cyber Security in IoT Networks," *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2019, pp. 0452-0457, doi: 10.1109/CCWC.2019.8666588.

[14] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," *arXiv preprint arXiv:1905.03554*, 2019.

[15] Y. Qin, J. Wei and W. Yang, "Deep Learning Based Anomaly Detection Scheme in Software-Defined Networking," *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Matsue, Japan, 2019, pp. 1-4, doi: 10.23919/APNOMS.2019.8892873.

[16] J. Jin, A. Dundar and E. Culurciello, "Flattened Convolutional Neural Networks for Feedforward Acceleration", in *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.

[17] P. Lin, K. Ye, and C. Z. Xu, "Dynamic network anomaly detection system by using deep learning techniques," in *International Conference on Cloud Computing*. Springer, 2019, pp. 161–176.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with DeepConvolutional Neural Networks, " in *Commun. ACM60.6*, pp. 84–90, 2017.

[19] J. Patterson and A. Gibson, *Deep Learning: A Practitioner's Approach*. O'Reilly Media, Inc, 2017.

[20] L. Zhou, M. Liao, C. Yuan and H. Zhang, "Low-Rate DDoS Attack Detection Using Expectation of Packet Size," *Security and Communication Networks*, vol. 2017, pp. 1-14, 2017.

[21] "Ogglas/Orignal-Slowloris-HTTP-DoS", *GitHub*, 2015. [Online]. Available: https://github.com/Ogglas/Orignal-Slowloris-HTTP-DoS. [Accessed: 03- Feb- 2020].

[22] "gkbrk/slowloris", *GitHub*, 2019. [Online]. Available: https://github.com/gkbrk/slowloris. [Accessed: 03- Feb- 2020].

[23] "adrianchifor/pyslowloris", *GitHub*, 2018. [Online]. Available: https://github.com/adrianchifor/pyslowloris. [Accessed: 03- Feb- 2020].

[24] "Gu1nness/SlowLoris", *GitHub*, 2017. [Online]. Available: https://github.com/Gu1nness/SlowLoris. [Accessed: 03- Feb- 2020].

[25] "ProjectMayhem/PySlowLoris", *GitHub*, 2017. [Online]. Available: https://github.com/ProjectMayhem/PySlowLoris. [Accessed: 03- Feb- 2020].

[26] M. G. Pecht and M. Kang. "Machine Learning: Data Pre-processing," in *Prognostics and Health Management of Electronics: Fundamentals, Machine Learning, and the Internet of Things*. IEEE, 2019.

[27] 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," *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, Montreal, QC, 2018, pp. 202-206, doi: 10.1109/NETSOFT.2018.8460090.

[28] S. Boukria and M. Guerroumi, "Intrusion detection system for SDN network using deep learning approach," *2019 International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS)*, Skikda, Algeria, 2019, pp. 1-6, doi: 10.1109/ICTAACS48474.2019.8988138.

[29] L. Bontemps, V. L. Cao, J. Mcdermott, and N. A. Le-Khac, "Collective anomaly detection based on long short-term memory recurrent neural networks," in *Proc. Int. Conf. Future Data Secur. Eng.*, 2017, pp. 141–152.

[30] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi and M. Ghogho, "Deep learning approach for Network Intrusion Detection in Software Defined Networking," *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, Fez, 2016, pp. 258-263, doi: 10.1109/WINCOM.2016.7777224.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," in *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. [Accessed 16 July 2020].

[32] S. Maeda, A. Kanai, S. Tanimoto, T. Hatashima and K. Ohkubo, "A Botnet Detection Method on SDN using Deep Learning," *2019 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, 2019, pp. 1-6, doi: 10.1109/ICCE.2019.8662080.

[33] C. Li et al., "Detection and defense of DDoS attack-based on deep learning in OpenFlow-based SDN," in *Int. J. Commun. Syst.*, vol. 31, no. 5, 2018.

[34] S. Chen, Y. Chen and W. Tzeng, "Effective Botnet Detection Through Neural Networks on Convolutional Features," *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, New York, NY, 2018, pp. 372-378, doi: 10.1109/TrustCom/BigDataSE.2018.00062.

[35] M. Ferrag, L. Maglaras, S. Moschoyiannis and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," in *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020. Available: 10.1016/j.jisa.2019.102419.