International Journal of Scientific Research in Science, Engineering and Technology
Print ISSN: 2395-1990 | Online ISSN : 2394-4099 (www.ijsrset.com)
doi : https://doi.org/10.32628/IJSRSET

Convolution Neural Network Machine Learning Algorithm Prediction Model for Intrusion Detection

Prof. Sapna Jain Choudhary, Noor Un Nihar
Shri Ram Group of Institutions, Jabalpur, Madhya Pradesh, India

Article Info
Volume 9, Issue 4

Page Number : 210-216

Publication Issue :
July-August-2022

Article History
Accepted : 05 July 2022
Published: 24 July 2022

ABSTRACT

Software Defined Networking (SDN) is evolving as a brand-new approach to the growth and innovation of the Internet. Since SDN can offer controllable, dynamic, and affordable networking, it is anticipated to be the Internet's ideal future. A rare chance to achieve network security in a more effective and flexible way is presented by the introduction of SDN. Because it has centralised control, SDN has the advantage of better network security provisioning as compared to traditional networks. However, in order to increase SDN security, it is necessary to address a number of additional network security challenges brought about by the SDN architecture's flexibility. The centralised controller, the control-data interface, and the control-application interfaces are the SDN's original structural weaknesses. Intruders may take advantage of these weaknesses.to conduct several types of attacks.

A crucial component of network architecture known as the Network Intrusion Detection System (NIDS) is utilised to identify network intrusions and secure the entire network. Using Deep Learning (DL) methods, we suggest an SDN-based NIDS (DeepIDS) in this thesis to look for anomalies in the SDN architecture. First, using various flow features, we assess the capability of DL for flow-based anomaly identification.

We demonstrate through studies that the DL technique has the capacity to detect flow-based anomalies in the SDN context. We also suggest a Gated Recurrent Unit Recurrent Neural Network to boost DeepIDS's detection rate. Our experimental findings demonstrate that the suggested model considerably increases the detection rate without degrading network performance. The effectiveness of our system in terms of precision, throughput, latency, and resource utilisation demonstrates that DeepIDS does not negatively impact the OpenFlow controller's performance, making it a workable strategy.

Finally, we present an unsupervised method to address the issue of an unlabelled and unbalanced dataset. This method results in a significant reduction in processing time while producing a high detection rate. Through thorough experimental evaluations, we determine that our suggested strategy we conclude

that our proposed approach exhibits a strong potential for intrusion detection in the SDN environments.

Keywords : IDS , DL , ML , IDS, SDN, CNN.

I. INTRODUCTION

Motivation

Software Defined Networking (SDN) is a developing architecture that is dynamic, manageable, cost-effective, and adaptable, thus making it ideal for the high- bandwidth, dynamic nature of today's applications and networks. SDNs are currently being deployed in many network environments, from home and enterprise networks to data centers (e.g., IBM, Cisco, Google WAN B4 [9], Huawei carrier network [10]). As can be seen in Figure1.1, the SDN market has grown to more than a $9.5 billion market in 2019 and is predicted to continue to grow to $13.8 billion by 2021. The capabilities of SDN (e.g., logically centralized controller, blood and knife the use of convolutional neural network.

and global network overview) help to solve several security issues in a traditional network and bring the ability to control network traffic at a fine-grained level. However, the SDN architecture itself also introduces some new attack threats and security vulnerabilities. Kreutz et al. [11] introduced seven threat vectors in SDN. Several attacks can be conducted in the SDN architecture. For instance, Distributed Denial of Service (DDoS) attacks can overwhelm an SDN controller and a communication channel with artificial service calls. A Man-in-the-Middle attack can break links between the controller and the switches and claim control of the network.

Because of the wide variety of types of SDN deployments, SDN security is a serious concern and has recently been extensively researched - see [12] and [13] for more detail.

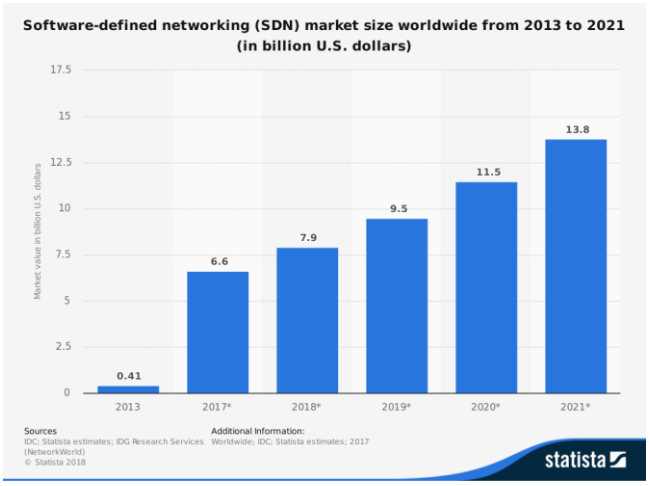


Figure 1.1: The SDN Market Size Prediction [1]

Therefore, there is a need to develop an efficient network intrusion

Challenges

In general, there are a few challenges for flow-based intrusion detection in the SDN architecture as follows: Network traffic is dynamic, diverse and constantly changing. In addition, network attacks keep evolving and become more intelligent and aggressive. The dynamic nature of network traffic makes intrusion detection extremely challenging.

Traditional NIDSs use a large number of hand-crafted features to improve intrusion detection accuracy. However, an SDN provides us with a limited number of raw flow features. Therefore, it is a challenge to improve intrusion detection accuracy with these limited raw flow features.

NIDS is supposed to provide real-time intrusion detection and mitigation. Therefore, computational

complexity and network overheads also need to be seriously considered.

For ML/DL intrusion detection approaches, the network data is signifi- cantly important. These datasets are used for training and testing systems for intrusion detection. The availability of labelled network datasets is a big problem. The SDN architecture is still a new technology, so network datasets for it are either quite rare and/or unpublished. As a result, it is difficult to train and evaluate a model in a supervised manner to detect intrusions in the SDN network.

II. RELATED WORK

The idea of "programmable networks" has been proposed as a way to facilitate the evolution of current networks. The concept of programmable networks and decoupled control logic has been around for several years. In the past, vari- ous technologies were developed to enable the programmability of communication networks. In the mid-1990s, Active Networks (AN) [24] were developed with the basic idea of injecting program into data packets. Switches extract and execute programs from data packets so that new routing mechanisms and network ser- vices can be implemented without the modification of the forwarding hardware. However, AN did not gain much attention because of its security and performance concerns. Also in mid 1990s, Devolved Control of ATM Network (DCAN) [25] was aimed at designing and developing the necessary infrastructure for scalable control and management of ATM networks. The premise of DCAN is that con- trol/management functions of the various network devices (e.g., ATM switches)

should be decoupled from the device themselves and delegated to external entities dedicated to that purpose. In the first half of the 2000s, the separation of the con- trol and data plane had been considered as one of the central points of simplifying network design. Forwarding and Control Element Separation (ForCES) [26] is a pioneer in this area. ForCES classified the network components into two distinct types which are the forwarding element and the control element. The forwarding element only forwards and filters traffic. The control plane provides instructions for processing packets. These elements communicate with each other via a standardised open interface, which is considered to be a core feature of the ForCES protocol. Although ForCES is still under active development, it is not widely adopted by major vendors. The IETF Network Configuration Working Group proposed NETCONF [27], which is a management protocol for modifying the configurations of network devices, in 2006. Network devices can expose an API that helps to send and to retrieve extensible configuration data. However, there is no separation between control and data planes. In 2006, the SANE/ Ethane project [28] proposed a new architecture for enterprise networks. Ethane focuses on using a centralized controller to manage policy and security in a network. It can be said that Ethane is the predecessor and the foundation for what would become SDN today.

SDN is defined by the Open Networking Foundation (ONF) which was founded in 2011 by Microsoft, Google, Facebook, Yahoo, Verizon and Deutsche Telekom. As of 2015, the organization has more than 150 industry members and receives endorsement by several network equipment vendors such as Cisco, Dell, Brocade and HP. An SDN architecture decouples the network control and forwarding functions enabling the network control to become directly programmable. The separation of the control plane from the data plane lays the ground for the SDN architecture. Network switches become simple forwarding devices, and the control logic is implemented in a physical or logical centralized controller.

The SDN concept was initially designed with significant advantages over the tra- ditional network. One of the crucial benefits of SDN is to make the highly vul- nerable traditional network more secure and robust. By centralizing the control plane, SDN

considerably simplifies the way that we integrate security mechanisms into our network. The evolution of networking brings several advantages, but it also brings the development of the network attacks. Attacks can be initiated from malicious management applications, the controller, and compromised hosts or switches. The main causes of concern lie in the SDN's main benefits: network programmability and control logic centralization. These capabilities introduce new faults and attack planes, which open the doors for new threats that did not exist before or are harder to exploit. The security issues of SDNs has been re- searched extensively in [11], [12], [13] and [55]. According to Kreutz et al. [12].

III. PROPOSED WORK AND RESULTS

Implementation of the 5 steps in the convolution neural network model life-cycle in Keras that we are going to look at.

1. Define Network.
2. Compile Network.s
3. Fit Network.
4. Evaluate Network.
5.Make Predictions.

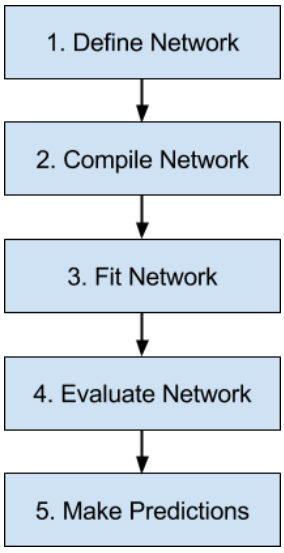


Figure 4.3 Life-Cycle for Convolution Neural Network Models using Keras

Input:
Software-Defined Network(SDN) dataset:
Step 1. Define Network:
The first step is to define your neural network. Neural networks are defined in Keras as a sequence of layers. The container for these layers is the Sequential class.
Step 2. Compile Network:
Once we have defined our network, we must compile it. Compilation is an efficiency step. It transforms the simple sequence of layers that we defined into a highly efficient series of matrix transforms in a format intended to be executed on your GPU or CPU, depending on how Keras is configured.
Think of compilation as a precompute step for your network. Compilation is always required after defining a model. This includes both before training it using an optimization scheme as well as loading a set of pre-trained weights from a save file. The reason is that the compilation step prepares an efficient representation of the network that is also required to make predictions on your hardware.
Compilation requires a number of parameters to be specified, specifically tailored to training your network. Specifically the optimization algorithm to use to train the network and the loss function used to evaluate the network that is minimized by the optimization algorithm.
Step 3. Fit Network:
Once the network is compiled, it can be fit, which means adapt the weights on a training dataset. Fitting the network requires the training data to be specified, both a matrix of input patterns X and an array of matching output patterns y.

The network is trained using the backpropagation algorithm and optimized according to the optimization algorithm and loss function specified when compiling the model. The backpropagation algorithm requires that the network be trained for a specified number of epochs or exposures to the training dataset.

Each epoch can be partitioned into groups of input-output pattern pairs called batches. This define the number of patterns that the network is exposed to before the weights are updated within an epoch. It is also an efficiency optimization, ensuring that not too many input patterns are loaded into memory at a time.

Step 4. Evaluate Network:

Once the network is trained, it can be evaluated. The network can be evaluated on the training data, but this will not provide a useful indication of the performance of the network as a predictive model, as it has seen all of this data before.

We can evaluate the performance of the network on a separate dataset, unseen during testing. This will provide an estimate of the performance of the network at making predictions for unseen data in the future.

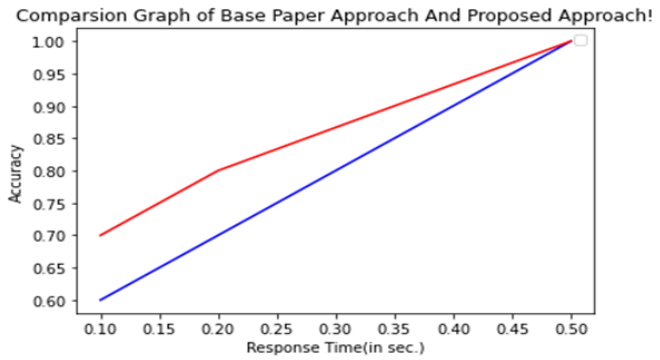
The model evaluates the loss across all of the test patterns, as well as any other metrics specified when the model was compiled, like classification accuracy. A list of evaluation metrics is returned.

Step 5. Make Predictions:

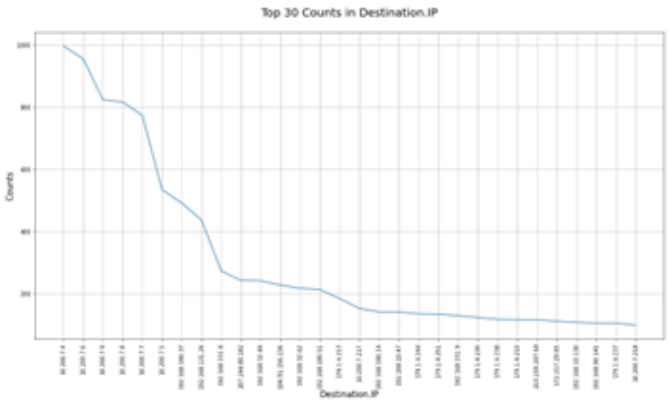
Finally, once we are satisfied with the performance of our fit model, we can use it to make predictions on new data.

This is as easy as calling the predict() function on the model with an array of new input patterns.

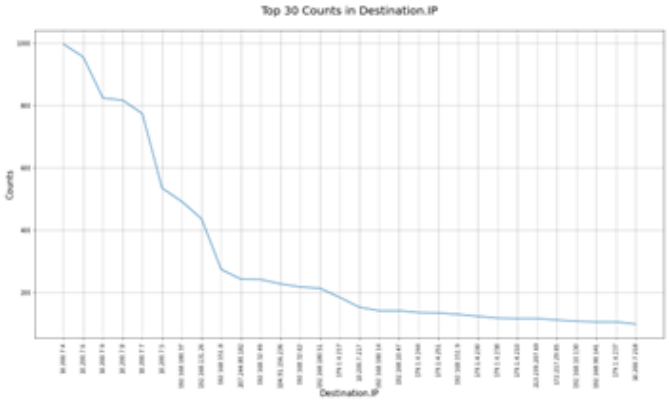
IV. RESULTS



(A) this shows the comparison on source IP and number of counts user hit on that IP address.



(B) This represents the comparison on Destination IP and number of counts user hit on that IP address.



(C) This represent comparison of our approach and base paper approach.

V. CONCLUSION AND FUTURE WORK

This work presents a framework that applies CNN algorithm is used as classifiers on the Software-Defined Network dataset. The experimental results show that, the best strategy is classifier getting an improved accuracy.

As mentioned in the previous chapters, SDN brings us a critical dilemma: an important potential evolution of networking architectures, along with a very dangerous increase in security problems. SDN introduces new faults, and attack planes that did not exist before or were harder to exploit.

These potential security issues are because of network programmability and control logic centralization in an SDN. However, an SDN can also be utilized to

strengthen network security. In this thesis, we have implemented an end-to-end NIDS - DeepIDS - for the SDN architecture. The DeepIDS can be deployed in any SDN and which then takes advantages of global network overview for intrusion detection.

VI. FUTURE WORKS

Currently, all the work has been done in an offline manner. All the DL algorithms are trained with several datasets to detect intrusion, but some of these datasets are outdated. In addition, some legitimate and anomaly traffic in these datasets are synthetic, so they cannot reflect real network scenarios. It would be better to implement our approach in an SDN testbed with real legitimate and anomaly traffic for further evaluation. It would be interesting to see how our method works with real networks and how quickly it can respond to the network.

VII. REFERENCES

[1]. J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," Electronics, vol. 9, p. 916, Jun. 2022.

[2]. R.PalanikumarandK.Ramasamy,"Softwaredefinednetworkbasedself- diagnosing faulty node detection scheme for surveillance applications," Comput. Commun., vol. 152, pp. 333–337, Feb. 2020.

[3]. Y. Goto, B. Ng, W. K. G. Seah, and Y. Takahashi, "Queueing analysis of software defined network with realistic OpenFlow–based switch model," Comput. Netw., vol. 164, Dec. 2019, Art. no. 106892.

[4]. A. Shaghaghi, M. A. Kaafar, R. Buyya, and S. Jha, "Software-defined network (SDN) data plane security: Issues, solutions, and future directions," in Handbook of Computer Networks and Cyber Security, B. Gupta, G. Perez, D. Agrawal, and D. Gupta, Eds. Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-030-22277-2_14.

[5]. K. Bhushan and B. B. Gupta, "Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud com- puting environment," J. Ambient Intell. Hum. Comput., vol. 10, no. 5, pp. 1985–1997, May 2019.

[6]. defined networks: A survey," IEEE Commun. Surveys Tuts., vol. 17, no. 4, pp. 2317–2346, 4th Quart., 2015.

[7]. H. Wang and W. Li, "DDosTC: A transformer-based network attack detection hybrid mechanism in SDN," Sensors, vol. 21, no. 15, p. 5047, Jul. 2021.

[8]. S.BoukriaandM.Guerroumi,"Intrusiondetection systemforSDNnet- work using deep learning approach," in Proc. Int. Conf. Theor. Applicative Aspects Comput. Sci. (ICTAACS), Dec. 2019, pp. 1–6.

[9]. Q. Niyaz, W. Sun, and A. Y. Javaid, "A deep learning based DDoS detection system in software-defined networking (SDN)," EAI Endorsed Trans. Secur. Saf., vol. 4, no. 12, p. e2, 2016.

[10]. 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," in Proc. Int. Conf. Wireless Netw. Mobile Commun. (WINCOM), Oct. 2016, pp. 258–263.

[11]. S. K. Dey and M. M. Rahman, "Flow based anomaly detection in soft- ware defined networking: A deep learning approach with feature selection method," in Proc. 4th Int. Conf. Electr. Eng. Inf. Commun. Technol. (iCEE- iCT), Sep. 2018, pp. 630–635.

[12]. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep recurrent neural network for intrusion detection in SDN-based networks," in Proc. 4th IEEE Conf. Netw.

Softwarization Workshops (Net- Soft), Jun. 2018, pp. 202–206.

[13]. J. Li, Z. Zhao, and R. Li, "Machine learning-based IDS for software- defined 5G network," IET Netw., vol. 7, no. 2, pp. 53–60, Mar. 2017. [14] P. Ding, J. Li, L. Wang, M. Wen, and Y. Guan, "HYBRID-CNN:

[14]. An efficient scheme for abnormal flow detection in the SDN-based smart grid," Secur. Commun. Netw., vol. 2020, pp. 1–20, Aug. 2020.

[15]. A.AbubakarandB.Pranggono,"Machinelearning basedintrusiondetec- tion system for software defined networks," in Proc. 7th Int. Conf. Emerg. Secur. Technol. (EST), Sep. 2017, pp. 138–143.

Cite this article as :

Prof. Sapna Jain Choudhary, Noor Un Nihar, "Convolution Neural Network Machine Learning Algorithm Prediction Model for Intrusion Detection", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 9 Issue 4, pp. 210-216, July-August 2022.
Journal URL : https://ijsrset.com/IJSRSET11229421