SAWANT: Smart Window based Anomaly Detection using Netflow Traffic

Mohammad Hashem Haghighat
Department of Automation
Tsinghua University
Beijing, China
l-a16@mails.tsinghua.edu.cn

Zohreh Abtahi Foroushani
Department of Automation
Tsinghua University
Beijing, China
zhou-yr18@mails.tsinghua.edu.cn

Jun Li
Research Institute of Information Technology
Tsinghua University
Beijing, China
junl@tsinghua.edu.cn

***Abstract*— Network security becomes a big concern nowadays. Although many solutions have been developed to detect network anomalies, the number of successful attacks like DDoS, Phishing, and Spam are boosting dramatically.**

In this paper, a novel behavior-based method, called SAWANT, is proposed to detect malicious rate of network traffic. SAWANT uses deep learning architecture to analyze netflow data, in which several meaningful attributes are extracted using a sliding window technique. Extracted attributes are then taken to a deep learning structure to identify malicious rate of each window, that represents the rate of abnormal netflow records per the window size. Experimental results over the well-known labeled botnet traffic CTU 13 showed that SAWANT was highly accurate as more than 99% of predicted malicious rates were correct, while it required a very small number of records for training.

Keywords- Deep Learning; Neural Network; Netflow; Network Intrusion Detection System; Pearson Correlation Coefficient

I. INTRODUCTION

Due to the rapid development of network technology, more and more systems tent to rely on network. However, the number of malicious activities such as DDoS, Man in the Middle, Phishing, and Eavesdropping attacks are increasing radically [1-3]. Hence, security has become more mission critical than ever in computer networks.

Network Intrusion Detection Systems (NIDS) protect networks from malicious attacks. Generally, NIDS is categorized into either signature-based or behavior-based detection systems. A signature-based NIDS compares the monitored traffics with the known patterns (attack signatures) to detect malicious activities [4-8]. Although signature-based technique provides the ability to detect already known attacks with no false negative, it is totally unable to find zero-day attacks. On the other hand, behavior-based approach is able to detect new attacks, by modeling network behavior [9-15]. However, this approach raises false positives, therefore minimizing false alarms is its main challenge.

For many years, machine learning algorithms were employed as conventional techniques to implement behavior-based detection engines. In order to detect abnormal behavior, a set of guidelines of how to use machine learning approaches were provided by Sommer et al. in [16].

Nowadays, deep learning has achieved great success and has been used widely in computer science for natural language processing, image and voice recognition. It provides the ability to find a correlation of analyzed data automatically. Therefore, this approach is a good option to be leveraged for NIDSes.

In this paper, we propose a smart window based deep learning framework, called SAWANT, to detect network anomalies. The key novelty of our approach is embedding network characteristics into deep learning procedure, where SAWANT analyzes the behavior of network traffic by aggregating flow records inside a window.

Most of the state-of-the-art techniques analyze packet headers or payloads, netflow records, or application layer logs to identify malicious activities. However, several network anomalies can only be revealed when these data are inspected together, such as Port/Range Scan, DDoS attacks, or Botnet C&C communications. As a result, considering batch records provides the ability to efficiently detect abnormal behavior with a very small set of training data. According to our experimental results, there is no need to train SAWANT with high number of netflow data as it achieved 99.952% detection rate using only one percent netflow records for training, which is magnificent.

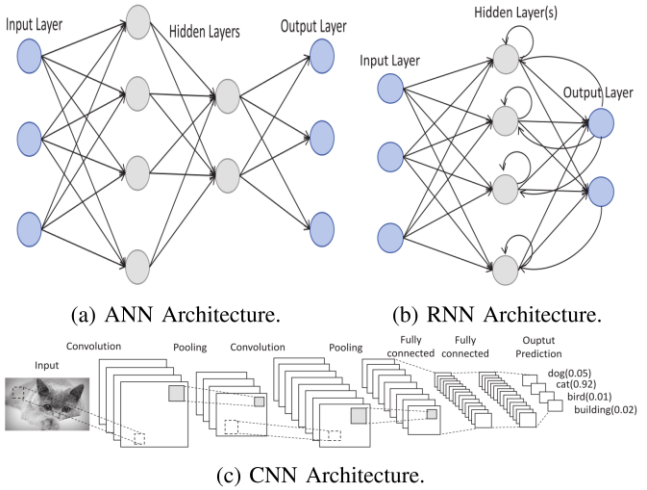
The rest of the paper is organized as follows: In section II, a review of deep learning-based IDSes are explained. Section III demonstrates SAWANT method step by step. The experimental results are presented in section IV. Finally, the paper is concluded and future research directions are provided in section V.

II. DEEP LEARNING-BASED NIDSES

In traditional machine learning techniques, feature extraction is manually applied and the system automatically learns how to map the achieved features to the outputs. However, in deep learning, multiple levels of features are automatically discovered and analyzed together to achieve the output [17]. There are various types of deep learning architectures including Deep Artificial Neural Network (ANN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Deep Belief Network (DBN) and so on.

ANN emulates human brain and its concepts in hierarchical architecture. It consists of input, output, and a set of hidden layers, where each layer comprises a set of learning units called neurons [18]. CNN is another class of

deep learning mostly used for analyzing images. Compared to ANN, it requires less variables to train the model, so that its learning procedure is much easier [19]. RNN is a type of deep learning originally designed for time series data, as it uses its internal states to processes sequence of inputs [20]. The key advantage of RNN is its ability of self-learning from the previous events. Fig. 1 compares different deep learning models.



(a) ANN Architecture. (b) RNN Architecture.

(c) CNN Architecture.

Figure 1. The comparison between ANN, CNN, and RNN [21].

Roy et al. [22] used deep ANN model to design an NIDS, which its accuracy was improved by multilayer feed-forward Neural Network with 400 hidden layer neurons. Rectifier and softmax activation functions are used for the output layer, and the model was evaluated by KDD Cup 99 dataset, with 41 features and around 4,900,000 connection records. The method achieved better results compared to SVM, using 75% of the dataset as training, and the rest for testing.

Li et al. [23] applied CNN for feature extraction and classification tasks. The main challenge of using CNN in NIDS is the conversion step. Although CNN ideally classify images, there is still a big challenge for text classification. In this work, a new conversion algorithm was proposed to map 41 features of NSL-KDD into 464 binary vectors. Then 464 vectors ware converted into 8*8-pixel images. So that the images were sent to CNN as training the model. Two learned CNN models, ResNet 50 and GoogLeNet were used for their experiments. The proposed CNN-IDS achieved accuracy of 77.14% and 79.14% using GoogLeNet and ResNet 50, respectively. However, the accuracy was not comparable to other deep learning models.

Yin et al. [24] presented a deep learning model based on RNN for NIDS purpose. It is argued that traditional machine learning techniques are unable to detect massive intrusion efficiently. The proposed approach started with the pre-processing phase which contains two operations on NSL-KDD dataset: mapping string features to binary vectors and also conducted feature normalization. Then, the features were sent to training phase, and the generated model was applied for testing phase. The experimental results indicated that the accuracy of intrusion detection was 83.28% and 81.29% for the binary and multi-class classification, respectively. Comparison of the RNN method with the traditional machine learning algorithms like SVM showed that this approach outperformed those techniques.

Kim et al. in [25] also developed a generative model of Long Short-Term Memory Recurrent Neural Network (LSTM-RNN). The experiments revealed that increasing hidden layer size achieved better system accuracy. The LSTM model detected attacks when it was trained by a very small subset of data (around 1%). However, the model ended up with some benign records as malicious as well more than 10% false positive rate.

Staudemeyer [26] tested different network topologies of RNN for network traffic modeling as time series with KDD Cup 99 dataset for extracting training data. With various parameters and structures of an RNN, such as learning rate, the number of memory blocks and the cells per memory block, the RNN model is capable to achieve highly accurate results in case of modeling the system by a large number of records.

Javaid et al. [27] proposed a new NIDS approach STL, that contains two phases: Unsupervised Feature Learning (UFL) and Supervised Feature Learning (SFL). For the UFL phase, Sparse Autoencoder is used to extract features, while the softmax regression function was applied for the classification task as the SFL phase. Using the NSL-KDD dataset, the STL method achieved 88.39% and 79.10% accuracy for 2-class and 5-class classifications, respectively, which is not comparable to RNN.

All the aforementioned methods took the advantage of deep learning structure to analyze network behavior. Although these methods achieved high accurate results over NSL-KDD dataset, they only analyze the original network records. We argue that aggregating the network records provides a complete overview of the network, and thus enables the NIDS to detect more advanced attacks with less false alarms.

Maimo et. al. in [28] aggregated netflow records using sliding window algorithm and generated a new dataset with 144 different attributes. A simple ANN as well as an advanced LSTM were employed for attack symptom and network anomaly detection, respectively. However, the accuracy of the method over CTU 13 dataset was not impressive as the F1-score was 89.4% in the best case due to the procedure of assigning the malicious label to each window.

To embed network characteristics into deep learning procedure, we proposed SAWANT as a sliding window based deep learning method. SAWANT aggregates netflow data in which several network attributes are computed. We also defined malicious rate as the label of each window, describing how many malicious records are presented inside a window.

III. SAWANT TECHNIQUE

To detect network anomalies, we proposed a novel deep learning technique called SAWANT (SmArt Window based Anomaly detection using Network Traffic). As depicted in Fig. 2, a pre-processing procedure is performed over netflow records to aggregate data and extract meaningful network

attributes. Hereafter, the extracted vectors are sent to a deep ANN to train the model. The output of SAWANT model is called malicious rate, in which identifies how much the aggregated vector is abnormal. In other words, malicious rate represents the number of anomalies inside an aggregated record.

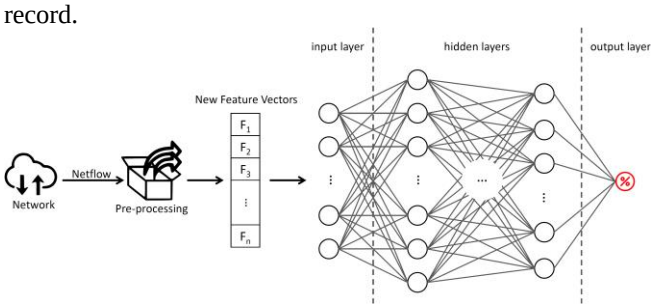


Figure 2. SAWANT Architecture.

A. Pre-processing

In order to maximize the accuracy of each deep learning model, it is necessary to understand the environment, first. Netflow records contain source/destination IP addresses and port numbers, protocol, flow duration, timestamp, flow size, number of packets level 3 protocol, next hop router, input and output SNMP interface, and TCP flags. These records are too simple to train a deep learning model, in which a few features will be extracted. Most of the network attacks have almost the same netflow attributes like Port/Range Scan, Botnet C&C Communications, DDoS Attacks and so on [11]. An aggregated view of the network traffic provides the ability to learn features and detect anomalies more accurate. The following procedure is defined to aggregate netflow records.

1. Slide a window of size *w* through the netflow records as illustrated in Fig. 3.

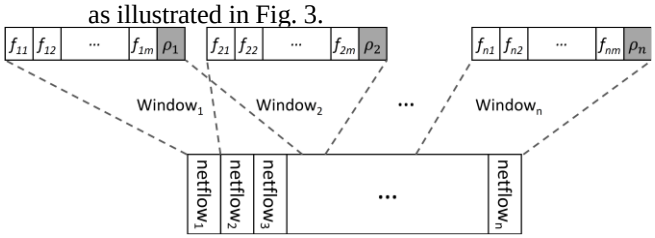


Figure 3. SAWANT Window based Feature Extraction Procedure.

2. For each position of the window compute the following features:
   - Number of unique values of Source IP, Source Port, Destination IP, Destination port, Duration, Source Bytes, Number of Packets, and Flow Size per incoming and outgoing flows.
   - Entropy values of Source IP, Source Port, Destination IP, Destination port, Duration, Source Bytes, Number of Packets, and Flow Size per incoming and outgoing flows.
   - Minimum, Maximum, Average, Sum, and Variance of Duration, Source Bytes, Number of Packets, and Flow Size per incoming, outgoing, and total flows.

   The features are explained in Table I.
3. Calculate malicious rate (named as ρ) of the window as the label of each feature vector according to the following equation:

$$\rho = \frac{\texttt{number of malicious netflow records}}{\texttt{Window Size}} \quad (1)$$

4. Use the achieved feature vector set and their labels (ρ) as the input data of ANN.

B. Deep Model

An ANN structure is considered as the deep learning model to analyze the new feature vectors. The output layer represents the predicted malicious rate, meaning the rate of abnormal flows inside corresponding window. The results of test dataset are compared with the actual malicious rate values using "Pearson Correlation Coefficient" function [29] to determine how accurate the predicted rates are.

In mathematics, Pearson Correlation Coefficient is defined as a measure of linear correlation between two different variables. The result is a value in range of [-1, 1], in which -1 means considered variables are negatively related, where increasing each of them results in decreasing the other. Respectively, 1 means both variables are positively related, so that, both variables are raised and dropped together. Finally, 0 means no relation between variables.

Definition 1. Let *X* and *Y* be two different variable sets. Pearson Correlation Coefficient "*r*" is computed according to the following equation:

$$\begin{aligned} r_{X,Y} &= \frac{E[XY] - E[X]E[Y]}{\sqrt{E[X^2] - E[X]^2}\sqrt{E[Y^2] - E[Y]^2}} \\ &= \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{\sqrt{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}\sqrt{\sum_{i=1}^{n} y_i^2 - n\bar{y}^2}} \end{aligned} \quad (2)$$

IV. SAWANT Evaluation

The SAWANT method is evaluated by a rich labeled dataset called CTU 13. We employed a core i7-6700HQ computer with 32GB memory to conduct our tests.

A. CTU 13 Dataset

CTU 13 is a labeled dataset captured from CTU University, Czech Republic, in 2011 [30]. It contains 13 days of botnet traffic including around 20 million netflow records and contains IRC, P2P, HTTP, Fast Flux, Spam, Click Fraud, Port Scan, and DDoS traffic.

B. Testbed Setup

In order to highlight the impact of deep learning hyper parameters on system's accuracy and speed, we observed SAWANT in various situations, as noted by Table II.

Three different ANN models having one, two, and four hidden layers, with the size of 100, 100-500, and 100-500-500-100 neurons were deployed to evaluate our model. In addition, a very small subset of pre-processed data was considered for training, while the rest were taken for testing. Moreover, the window size of 1000, with step values of 1,

TABLE I. EXTRACTED FEATURES.

	# Unique Values	Entropy	Min	Max	Mean	Sum	Variance	Total
Source IP	2[a]	2	-	-	-	-	-	4
Source Port	2	2	-	-	-	-	-	4
Destination IP	2	2	-	-	-	-	-	4
Destination Port	2	2	-	-	-	-	-	4
Duration	2	2	3	3	3	3	3	19
# Packets	2	2	3	3	3	3	3	19
Source Bytes	2	2	3	3	3	3	3	19
Flow Size	2	2	3	3	3	3	3	19
Total	16	16	12	12	12	12	12	92

a. Each cell represents the number of features.

TABLE II. SAWANT ANN MODEL.

Hyper Parameters	Values
Hidden Layers	1, 2, 4
layer Size	(100), (100, 500), (100, 500, 500, 100)
Activation Function	Rectified Linear Unit (Relu)
Batch Input	Off, On
Dropout	0, 0.2
Train Size	10%, 5%, 2%, 1%
Test Size	90%, 95%, 98%, 99%
Window Size	1000
Step	1, 10, 100

10, and 100 were employed as the sliding window mechanism. In what follows, the evaluation results are explained.

C. *Correlation Coefficient*

We performed the test using the hyper parameters provided by Table II. The result was predicted malicious rate in which we compared it with the actual value using Correlation Coefficient function. Fig. 4 shows an example of predicted malicious rates compared by the actual values using the configured parameters explained by Table III, with the Correlation Coefficient value of 0.99382.

TABLE III. AN EXAMPLE OF PREDICTED MALICIOUS RATE.

Hyper Parameters	Values
Hidden Layers	4
layer Size	(100, 500, 500, 100)
Activation Function	Rectified Linear Unit (Relu)
Batch Input	On
Dropout	0.2
Train Size	10%
Test Size	90%
Window Size	1000
Step	1

As mentioned earlier, CTU 13 has about 20 million netflow records. We deployed one, two, five, and ten percent of pre-processed data to the system in order to train the model, which the size of the training sets are described by Table IV.

TABLE IV. SIZE OF TRAINING SET.

Train Size \ Step	*1*	*10*	*100*
1%	196288	19628	1962
2%	392577	39257	3925
5%	981443	98144	9814
10%	1962886	196288	19628

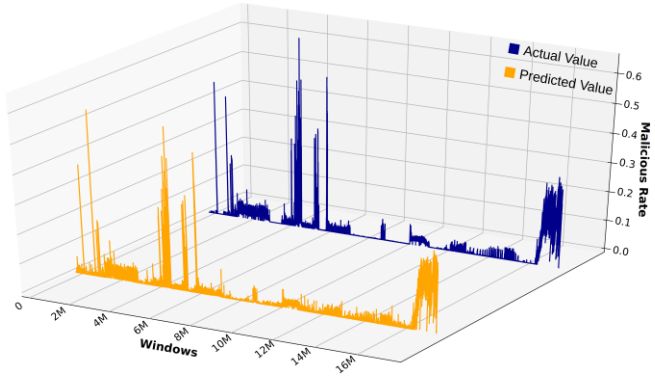


Figure 4. Predicted Malicious Rate Vs. Actual Malicious Rate.

The detailed correlation coefficient results are expressed by Table V.

As highlighted in Table V, leveraging bigger training sets as well as choosing 4-layers ANNs provided more detection rate, which is illustrated in Fig. 5.

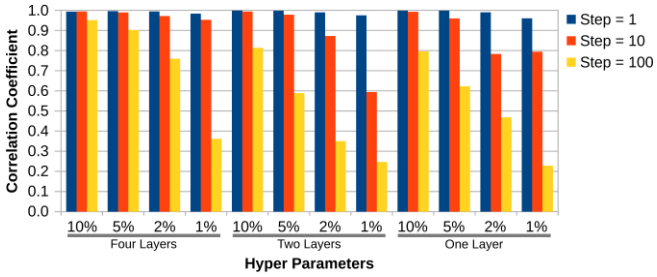


Figure 5. Correlation Coefficient for different step values, number of layers, and training size, where batch and Dropout were set to "On" and "0.2", respectively.

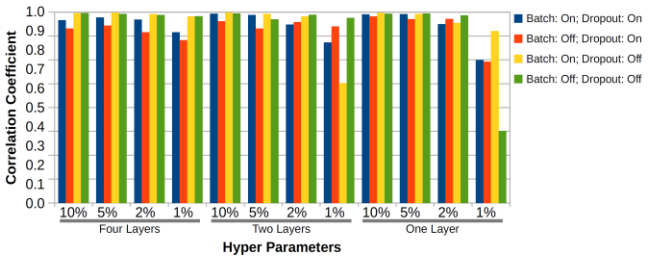
Besides, batch input and dropout function plays an important role to avoid over fitting. As highlighted in Fig. 6, feeding the model in batch mode accompanying with dropping random neurons, result in more detection rate in most cases.

D. *Training Time*

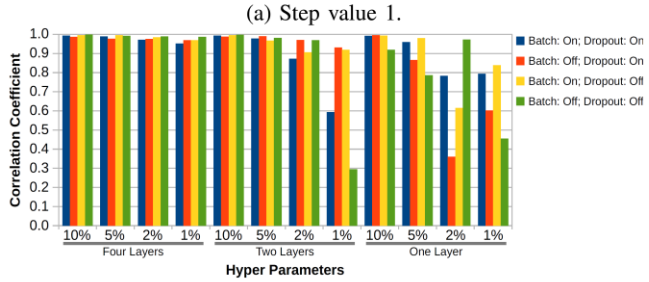
System training time is a key factor of deep learning environments. Dozens of parameters impact the training speed of any deep learning models, but batch input influences one of the most. Fig. 7 highlights the effect of

TABLE V. CORRELATION COEFFICIENT RESULTS.

Dropout				On						Off					
Batch Input				On			Off			On			Off		
Step				1	10	100	1	10	100	1	10	100	1	10	100
Number of Layers	4	Train Size	10%	0.99	0.99	0.95	0.98	0.98	0.96	0.99	0.99	0.96	0.99	0.99	0.98
			5%	0.99	0.98	0.90	0.98	0.97	0.95	0.99	0.99	0.92	0.99	0.99	0.96
			2%	0.99	0.97	0.75	0.98	0.97	0.92	0.99	0.98	0.84	0.99	0.98	0.94
			1%	0.98	0.95	0.36	0.97	0.96	0.89	0.99	0.96	0.50	0.98	0.98	0.89
	2	Train Size	10%	0.99	0.99	0.81	0.99	0.98	0.80	0.99	0.99	0.92	0.99	0.99	0.81
			5%	0.99	0.97	0.58	0.98	0.99	0.92	0.99	0.96	0.83	0.98	0.98	0.94
			2%	0.98	0.87	0.35	0.99	0.97	0.34	0.99	0.90	0.60	0.96	0.96	0.91
			1%	0.97	0.59	0.24	0.98	0.93	0.63	0.94	0.91	0.31	0.29	0.29	0.82
	1	Train Size	10%	0.99	0.99	0.79	0.99	0.99	0.82	0.99	0.99	0.90	0.91	0.91	0.95
			5%	0.99	0.95	0.62	0.99	0.86	0.66	0.99	0.98	0.90	0.78	0.78	0.85
			2%	0.98	0.78	0.46	0.99	0.36	0.25	0.99	0.61	0.41	0.97	0.97	0.44
			1%	0.95	0.79	0.22	0.95	0.60	0.34	0.98	0.83	0.50	0.45	0.45	0.68



(a) Step value 1.



(b) Step value 10.

Figure 6. The key role of batch input and dropout function in deep learning.

using batch input in our model, where using batch input incredibly raised the speed up to five times.

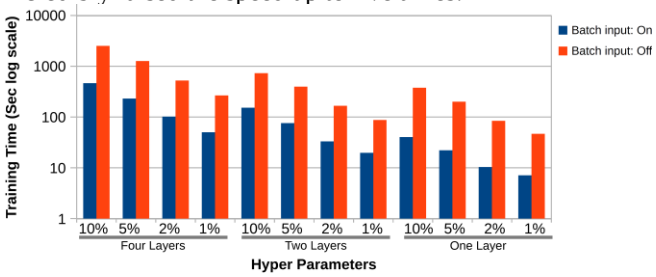


Figure 7. Batch input On Vs. Off.

In contrast to batch input, using dropout slightly decreased the speed, as the system trained dataset, in about 343 seconds using a four-hidden layers ANN, having no dropout configuration, while using this function raised the training time about 30%. Fig. 8 depicts the dropout impact on the system training time.

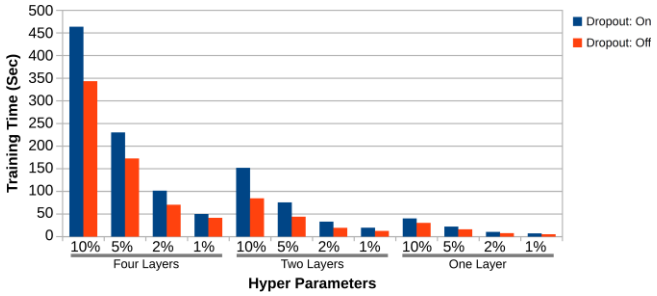


Figure 8. Dropout input On Vs. Off.

Table VI summarize the training time of SAWANT using different configurations.

V. CONCLUSION

This paper proposed SAWANT, a novel deep learning window-based technique to predict malicious rate of the windows. Analyzing the whole network traffic is neither applicable nor practical in deep learning area due to user privacy issue and huge training and testing time, hence SAWANT used netflow traffic.

Experimental results revealed that SAWANT provided high accurate malicious rate detection (99.952%) for a well-known labeled traffic named CTU 13, while it was trained by a very small number of records.

Although SAWANT provided high accurate prediction for malicious rate of each window, it is necessary to predict netflow labels. In the future we plan to identify the flow labels by observing the predicted malicious rates. Also using more advanced deep learning structures like RNN or DBN is another direction for the future.

ACKNOWLEDGMENT

This work was supported by the National Key Technology R&D Program of China under Grant No. 2015BAK34B00 and the National Key Research and Development Program of China under Grant No. 2016YFB1000102.

TABLE VI. SYSTEM LEARNING TIME.

Dropout				On						Off					
Batch Input				On			Off			On			Off		
Step				1	10	100	1	10	100	1	10	100	1	10	100
Number of Layers	4	Train Size	10%	463.7	42.04	5.25	2515.41	249.19	25.78	343.28	32.94	4.13	2269.25	219	21.97
			5%	230.17	21.48	3.25	1263.1	125.41	13.2	172.54	16.73	2.49	1145.07	107.03	11.83
			2%	101.54	9.1	1.81	521.39	48.92	6.52	70.33	7.05	1.33	474.67	41.8	5.09
			1%	50.05	4.99	1.53	265.43	25.1	3.58	41.36	3.88	1.13	248.01	20.79	3.29
	2	Train Size	10%	152.02	13.85	2.05	728.42	70.39	7.75	84.57	8.8	1.38	618.73	57.3	6.40
			5%	75.66	7.17	1.26	395.57	35.96	4.35	43.88	4.5	0.84	314.34	28.92	3.31
			2%	33.11	3.34	0.87	166.54	15.12	2.24	19.34	2.31	0.72	139.37	12.26	1.62
			1%	19.68	2.03	0.8	87.21	7.96	1.59	12.56	1.39	0.53	75.36	6.41	1.12
	1	Train Size	10%	40.25	4.25	0.8	377.82	37.65	0.66	30.39	3.23	4.13	336.31	32.82	3.68
			5%	22.12	2.33	0.57	200.16	20.04	0.46	15.98	1.87	2.32	181.61	17.34	2.13
			2%	10.34	1.2	0.45	84.3	8.09	0.45	7.81	0.95	1.65	77.31	7.19	1.05
			1%	7.11	0.84	0.43	46.53	4.29	0.37	5.12	0.74	0.82	42.44	3.87	0.80

REFERENCES

[1] Akamai, "2018 state of the internet / security: A year in review," 2018. [Online]. Available: https://www.akamai.com/us/en/multimedia/documents/state-of-theinternet/2018-state-of-the-internet-security-a-year-in-review.pdf

[2] ——, "2019 state of the internet / security: Ddos and application attacks," 2019. [Online]. Available: https://www.akamai.com/us/en/multimedia/documents/state-of-theinternet/state-of-the-internet-security-ddos-and-application-attacks-2019.pdf

[3] AoN, "2019 cyber security risk report, whats now and whats next," 2019. [Online]. Available: https://www.aon.com/getmedia/51bff3db-20ea-46dd-a9aa-1773cfe089ce/Cyber-Security-Risk-Report-2019.pdf.aspx

[4] A. Bakshi and Y. B. Dujodwala, "Securing cloud from ddos attacks using intrusion detection system in virtual machine," in Communication Software and Networks, 2010. ICCSN'10. Second International Conference on. IEEE, 2010, pp. 260–264.

[5] C. Douligeris and A. Mitrokotsa, "Ddos attacks and defense mechanisms: classification and state-of-the-art," Computer Networks, vol. 44, no. 5, pp. 643–666, 2004.

[6] J. Ioannidis and S. M. Bellovin, "Implementing pushback: Router-based defense against ddos attacks." in NDSS, 2002.

[7] A. M. Lonea, D. E. Popescu, and H. Tianfield, "Detecting ddos attacks in cloud computing environment," International Journal of Computers Communications & Control, vol. 8, no. 1, pp. 70–78, 2013.

[8] V. Vaidya, "Dynamic signature inspection-based network intrusion detection," Aug. 21 2001, uS Patent 6,279,113.

[9] S. Behal, K. Kumar, and M. Sachdeva, "D-face: An anomaly based distributed approach for early detection of ddos attacks and flash events," Journal of Network and Computer Applications, vol. 111, pp. 49–63, 2018.

[10] O. E. Elejla, B. Belaton, M. Anbar, and A. Alnajjar, "Intrusion detection systems of icmpv6-based ddos attacks," Neural Computing and Applications, vol. 30, no. 1, pp. 45–56, 2018.

[11] M. H. Haghighat and J. Li, "Edmund: Entropy based attack detection and mitigation engine using netflow data," in Proceedings of the 8th International Conference on Communication and Network Security. ACM, 2018, pp. 1–6.

[12] M. Idhammad, K. Afdel, and M. Belouch, "Semi-supervised machine learning approach for ddos detection," Applied Intelligence, vol. 48, no. 10, pp. 3193–3208, 2018.

[13] D. S. Terzi, R. Terzi, and S. Sagiroglu, "Big data analytics for network anomaly detection from netflow data," in Computer Science and Engineering (UBMK), 2017 International Conference on. IEEE, 2017, pp. 592–597.

[14] J. M. Vidal, A. L. S. Orozco, and L. J. G. Villalba, "Adaptive artificial immune networks for mitigating dos flooding attacks," Swarm and Evolutionary Computation, vol. 38, pp. 94–108, 2018.

[15] R. Wang, Z. Jia, and L. Ju, "An entropy-based distributed ddos detection mechanism in software-defined networking," in Trustcom/Big-DataSE/ISPA, 2015 IEEE, vol. 1. IEEE, 2015, pp. 310–317.

[16] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in 2010 IEEE symposium on security and privacy. IEEE, 2010, pp. 305–316.

[17] 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 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM). IEEE, 2016, pp. 258–263.

[18] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury et al., "Deep neural networks for acoustic modeling in speech recognition," IEEE Signal processing magazine, vol. 29, 2012.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[20] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013, pp. 6645–6649.

[21] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrows intelligent network traffic control systems," IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2432–2455, 2017.

[22] S. S. Roy, A. Mallik, R. Gulati, M. S. Obaidat, and P. Krishna, "A deep learning based artificial neural network approach for intrusion detection," in International Conference on Mathematics and Computing. Springer, 2017, pp. 44–53.

[23] Z. Li, Z. Qin, K. Huang, X. Yang, and S. Ye, "Intrusion detection using convolutional neural networks for representation learning," in International Conference on Neural Information Processing. Springer, 2017, pp. 858–866.

[24] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," Ieee Access, vol. 5, pp. 21 954–21 961, 2017.

[25] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in 2016

International Conference on Platform Technology and Service (PlatCon). IEEE, 2016, pp. 1–5.

[26] R. C. Staudemeyer, “Applying long short-term memory recurrent neural networks to intrusion detection,” South African Computer Journal, vol. 56, no. 1, pp. 136–154, 2015.

[27] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, “A deep learning approach for network intrusion detection system,” in Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS). ICST (Institute for Computer Sciences, Social-Informatics and , 2016, pp. 21–26.

[28] L. F. Maimo´, A´ . L. P. Go´mez, F. J. G. Clemente, M. G. Pe´rez, and G. M. P´erez, “A self-adaptive deep learning-based system for anomaly detection in 5g networks,” IEEE Access, vol. 6, pp. 7700–7712, 2018.

[29] J. Benesty, J. Chen, Y. Huang, and I. Cohen, “Pearson correlation coefficient,” in Noise reduction in speech processing. Springer, 2009, pp. 1–4.

[30] CTU, “Ctu-13 botnet traffic dataset,” 2011. [Online]. Available: https://mcfp.weebly.com/the-ctu-13-dataset-a-labeled-dataset-with-botnet-normal-and-background-traffic.html