International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Stacked Autoencoder Based Neural Network for Identifying Malicious Traffic in SDN

Aaditya Jain[1], Garima Jain[2], Dr.Pallavi R[3], Atul Dadhich[4]

Submitted:22/04/2023 **Revised:**14/06/2023 **Accepted:**25/06/2023

Abstract: Deep packet inspection (DPI) has drawn a lot of interest in software-defined networking (SDN) because sophisticated assaults might smuggle harmful payloads into packets. Third-party proprietary pattern-based or port-based DPI solutions may struggle to handle a large amount of data flow effectively. In order to provide adaptive and effective packet assessment, a unique stacked autoencoder based Convolutional Neural Network (SA-CNN) approach is described in this research. The first step in SA-CNN's early detection prescription is to scan each new flow's IP address through SA-CNN. Following that, SA-CNN enables profound packet assessment at the packet-level granularity: (i) for unencrypted packets, stacked autoencoder extract the features of reachable payloads, together with tri-gram incidence based on Term Frequency and Inverted Document Frequency (TF-IDF) and linguistic properties. These qualities are combined into a sparse matrix representation rather than matching with particular pattern combinations in order to train a CNN classifier. The SA-CNN presents an adaptive packet sampling window that utilizes linear prediction to balance the degree of detection precision as well as the bottleneck of the SDN controller; and (ii) for encrypted packets, the SA-CNN extracts salient features from packets and then trains a CNN classified with a another methods, slightly than decrypting the encrypted traffic to compromise user solitude. On the Mininet platform and Ryu SDN controller, an SA-CNN prototype is put into operation. Through experiments, the presentation and overhead of the suggested explanation are evaluated with datasets from the actual world. The arithmetical outcomes show that SA-CNN can significantly increase detection accuracy while maintaining reasonable overheads.

Keywords: *software-defined networking (SDN), stacked autoencoder based Convolutional Neural Network (SA-CNN), malicious, Term Frequency and Inverted Document Frequency (TF-IDF), deep packet inspection (DPI)*

1. Introduction

In software-defined networking architecture, the network is separated into a separate data plane and control plane. The SDN controller is an independent component that consolidates management of the network. A centralized view and level of control over the whole network are provided by the SDN controller. SDN enables network operators and managers to design and automate network behavior. The network can adapt and react quickly to changing needs and circumstances because to its programmability. By allowing dynamic traffic engineering and effective network resource usage, SDN improves network dependability. Based on network circumstances, the centralized control plane may divert traffic in real-time to prevent congestion or failures [1]. It is possible to apply the supervised learning algorithm SDM for both classification and regression problems. Finding the best hyper plane in feature space to optimally separate data points from various classes is the goal of SDM. SDM has a solid theoretical base and has shown outstanding performance in a number of fields. Coordinated Management Network administration is more consolidated and easier when the control plane is housed in the SDN controller. From a single point of management, administrators may design network rules, set up network components, and keep an eye on the network [2]. Network administrators may develop and implement network rules and settings using software thanks to SDN's programmability. Due to its programmability, network administration chores may be automated, which minimizes human labor and risk of mistake. Contrarily, conventional networks sometimes need for manual setup of individual devices, which, in large-scale deployments, may be time-consuming and error-prone. Through a different SDN controller, SDN makes it possible for centralized administration. It is simpler to install, administer, and monitor network devices and services because to this centralization, which offers a uniform view and management over the whole network infrastructure. Traditional networks often spread administration chores among various devices, which increases complexity and

1Assistant Professor, College of Computing Science and Information Technology, Teerthanker Mahaveer University, Moradabad, Uttar Pradesh, India, Email id: jain.aaditya58@gmail.com

2Assistant Professor & Dy. HoD, Department of Computer Science and Business Systems (CSBS), Noida Institute of Engineering and Technology, Greater Noida, Uttar Pradesh, India, Email id: garimajain@niet.co.in

3Professor & HOD, Department of Computer Science and Engineering, Presidency University, Bangalore, India, Email Id: pallavi.r@presidencyuniverstiy.in

4Assistant Professor, Department of Electrical Engineeing, Vivekananda Global University, Jaipur, India, Email Id- dadhich.atul@vgu.ac.in

raises the possibility of discrepancies [3]. Effective network segmentation, which is essential for boosting security, is made possible by SDN. SDN may stop lateral migration of threats and limit possible security breaches by logically isolating distinct areas of the network and managing access between them. Sensitive data may be isolated through network segmentation, which also helps to limit unwanted access and shrink the attack surface. Security regulations may be implemented and enforced from a single location thanks to SDN's centralized control plane. Administrators may design and administer security rules at the SDN controller, ensuring uniform policy enforcement across the network. This unified method streamlines security administration and lowers the possibility of incorrect setups or inconsistent rules [4]. SDN enables dynamic modifications and adjustments to network configurations and behavior. Network managers may programmatically define and change network behavior. Due to its dynamic nature, network resources may be quickly provisioned, reconfigured, and optimized in response to shifting circumstances or demands. SDN provides better management than conventional networking strategies. A single view and centralized administration of the network infrastructure are made possible by the centralized control plane offered by SDN. Network administration activities may be made simpler and less difficult by network managers being able to set and enforce rules, configure network devices, and monitor network performance from a single location [5]. The SDN controller is a distinct entity that unifies and decouples the control plane services. The whole network infrastructure is under the administration and control of the SDN controller. It interfaces with switches and other network hardware. Network managers may programmatically create and change network behavior using the SDN controller. Administrators may operate and configure network devices dynamically using APIs or programming interfaces, offering automation and flexibility. The SDN controller has a comprehensive understanding of the whole network. It gathers real-time data on device statuses, traffic patterns, and network architecture. This comprehensive perspective makes network administration, optimization, and troubleshooting effective [6]. The administration of policies is simplified by SDN. Administrators may design and maintain rules centrally using a centralized control plane, usually at the SDN controller. Because rules are configured centrally rather than on individual network devices, there is less complexity and the chance of mistakes. Network policies may be dynamically updated thanks to SDN. By changing the configuration at the SDN controller, policy changes may be put into effect immediately. The network devices are then updated with these changes, guaranteeing uniform policy enforcement throughout the whole network. Rapid adaptability to shifting requirements or security demands is made possible by this flexibility. This adaptability allows prompt responses to new security risks or evolving business requirements [7]. Social networks often include privacy options that let users choose who may see their material and interact with them. To restrict access to their sensitive information, users should be aware of these settings and modify them as desired. Social networks have a duty to safeguard the information that its users share with them. To protect sensitive information, they should put in place the proper security measures, including encryption, secure authentication methods, and frequent security audits. Many social networks enable access to user data by other programs. To prevent their sensitive information from being used improperly, users should carefully check the permissions provided to these programs and understand how their data will be used [8]. SDN is architecture that enabling network programming and central management. In SDN, the control plane, which deals with how network devices make decisions, is separated from the data plane, which controls how network traffic is sent. SDN does not necessarily entail a physical link between all network devices, despite the fact that it has several advantages, including enhanced flexibility, scalability, and automation. Whether a network uses SDN or conventional networking techniques, the physical connectivity is still accomplished via physical links like Ethernet cables or wireless connections [9]. The network flow data, such as packet headers or flow statistics, would be the input to the autoencoder, and the reconstructed flow data would be the output. The autoencoder may learn to encode and reconstruct the typical patterns of network traffic by being trained on a dataset of typical network flows. The autoencoder may be used to encode fresh network flow data into the learnt representation after being trained as a data representation model. For purposes of further analysis or categorization, such as the detection of abnormalities or malicious network traffic, this representation may be passed into another model or algorithm [10]. The characteristics of the malicious traffic are trying to identify. To adapt to new assault patterns and maintain the model's efficacy over time, regular updates and retraining are also necessary.

2. Related Works

There are several notable ways in which SDN differs from the conventional network, and these differences are what have piqued the interest of many. To alter the switch's regulations, SDN allows for its programmability. CNN and RNN are used to offer a technique for classifying harmful network traffic. Tensor Flow, a framework that supports graphics processing units (GPUs), is used to construct our suggested technique. We tested our methodology with three different data sets. The findings show that our concept has significant user potential in SDN security and

outperforms current methods in terms of detection accuracy and stability [11]. SDN is a solution that is gaining popularity. To detect unknown network assaults, NIDS utilizes anomalous traffic detection. Griffin, a per-packet anomaly detection system with dynamic neural network-based training model updates, is suggested in this study. In order to effectively separate aberrant traffic from regular traffic, The Griffins is implemented in an SDN setting. To modify the training model, the autoencoder are efficient depending on the origin indicate square error. Unsupervised adjustment means there is no requirement for a specialist to categorize complex traffic or regularly inform the reproduction [12]. The SDN paradigm is a developing design that may be used to create future networks and satisfy changing application requirements. A finding and protection scheme based on adversaral preparation in SDN has been suggested; it utilizes the Generative Adversarial Network (GAN) architecture to identify DDoS assaults and use adversarial training to reduce the system's sensitivity to adversarial attacks. With the help of clearly defined modules, the suggested system enables permanent traffic monitoring utilizing IP pour analysis, allowing the difference finding system to respond almost instantly [13]. A SDN in that they keep tabs on all network activity. To improve NIDS's resilience against adversarial example assaults, we propose using denoising autoencoders and a novel approach called reconstruction from partial observation (RePO) to identify many forms of complex attacks with a low false alarm rate. Their testing on a collection of real-world network assaults demonstrates that denoising autoencoders may enhance detection of malicious traffic [14]. A solution called SDN has emerged and been future as a way to improve the internet's underlying architecture. Shows how NIDS in an SDN controller may utilize machine learning methods to analyze network traffic for signs of malicious activity. In this example of attack detection, we use three distinct tree-based machine learning algorithms to show how effective each can be: Random Forest, XGBoost and Decision Tree is used [15].

3.Methodology

3.1. Early detection

SA-CNN has a basic early detection system in place before starting a thorough payload research, keeping in mind that IP tackle filter is still required, particularly for which partial payload assessment is worthless. F is the bidirectional connection between the nodes that the cars use to get from one place to another. Packets are the building blocks of a flow. Every packet has a five-tuple associated with it, which includes the protocol type, source IP address, purpose IP address, source port, and these four additional information.

The procedure of processing packets in SDN is shown in fig 1.

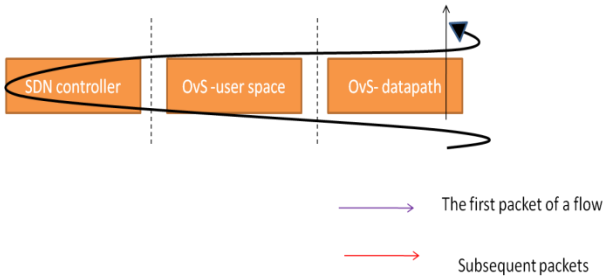

Fig.1. SDN packet workflow

The datapath module in the kernel space of the examines all external network packets and extracts key variables to validate against flow tables. If the datapath module is unable to find a matching entry, it will attach the packet to trigger upcall, which notifies the OvS user space that a message has arrived. If a packet arrives without a matching flow entry in user space, it will always trigger a Packet in message to be forwarded to the organizer. Packet-in messages are unnecessary if and only if subsequent packets match beside flow tables maintained in the kernel. In this article, the SDN controller has a blacklist of IP address accessible for monitoring the packet's source IP address for early detection. In either case, the SDN controller will implement the OvS's corresponding flow entries. Deep packet inspection is only necessary after a successful search of the flow table using the prior packet.

3.2. Unencrypted packets inspection

Deep packet inspection is able to access the payloads of the packets in unencrypted transmission. However, the system incurs significant costs in order to analyze all of the payloads. At order to solve the problem of resources and performance at packet-level granularity, this work develops linear prediction-based adaptive packet-driven sampling technique.

3.2.1 A packet sampling method based on LP

In order to do deep packet inspection, a packet sample window collects a series of packets in a row. The amount of packets that were sampled determines the packet window size. The fundamental idea behind LP-based adaptive packet sampling is to try to predict the packet window range of the subsequent example in order to collect successive $u_j$packets commencing each m packets in a flow, wherever $m \gg w$, i R. even that is resolute empirically. A linear mixture of the prior n sample is used to represent the expected sampled packets $\hat{\delta}_{m+1}$ in the (m + 1)th model:

$$\hat{\delta}_{m+1} = \delta_m + \frac{\Delta u}{m-1}\sum_{j-1}^{m-1}\left(\frac{\delta_{j+1}-\delta_j}{u_{j+1}-u_j}\right), n \geq 2 \qquad (1)$$

Where δ_j is the true numeral of malicious packets in the jth sample, and u is the existing fluctuation of the packet window dimension among u_{m+1} along with u_m, δ_{m+1} and ∂_m, computed as Equation (2-4):

$$\Delta u = u_{m+1} - u_m \quad (2)$$

$$\frac{\hat{\delta}_{m+1}-\delta_m}{\Delta u'} = \frac{\delta_{m+1}-\partial_m}{\Delta u} \quad (3)$$

$$Q(\hat{\delta}_{m+1}) = \frac{\hat{\delta}_{m+1}-\delta_m}{\delta_{m+1}-\delta_m} \quad (4)$$

In the following example, Eq. (5 and 6) depicts packet window size variation.

$$\Delta u' = \begin{cases} 5 \\ -\frac{1}{2} \times \Delta u \end{cases} \quad \begin{matrix} if\ u_{m+1} = u_m \\ otherwise \end{matrix} \quad (5)$$

$$\Delta u' = -\frac{\hat{\delta}_{m|+1}-\delta_{m+1}}{|\hat{\delta}_{m+1}-\delta_{m+1}|} \left| Q\left(\hat{\delta}_{m+1}\right).\Delta u \right| \quad (6)$$

A requirement is met by the range of values of the packet window that is not permitted to be exceeded. The difference among the expected value and the actual value enables the complex proprietor to pro-actively provide adaptive packet sampling by flexibly and promptly adjusting the packet window size.

3.2.2. Extracting features

The payloads are the parts of a packet that don't include packet headers and contain the data itself. Sometimes the payloads of the packets are character-based text strings. A payload is a string of letters made up of both alphabets and numerals. The machine learning method, however, is unable to interpret the payloads in the thread format directly. In order to gather data characteristics for further modeling, word embedding which converts an expression to a vector using a dictionary is used to overcome this problem.

In order to represent how crucial a given word is to a payload, the TF-IDF, which may be stated as follows, attempts to assign such frequent terms with lesser weights while increasing the relevance of individuals words important to a certain payload, which may be represented as:

$$TF--IDF = TF\ *IDF \quad (7)$$

Where T F is a payload's word frequency in terms of how often it appears. Larger IDF is often the outcome of a term that occurs seldom in a text, proving that this word is considerably better for payload categorization. Instead than handling encoded payload data directly, TF-IDF tries to extract numerical characteristics from payload strings, particularly feature vectors. Trigram is used to segment payloads and determine the TF-IDF value. A collection of all quality sequence having a distance end to end of three constitutes a trigram of the payload.

a. Linguistic Features

Using the digit count, harmful payloads may be identified and avoided. The quantity of digits is definite by the number of digit in the payload. The lengths of the recurring digit sequences are calculated by adding the intervals between each pair of consecutive digits.

Table 1. Linguistic traits as examples

Characteristics	***A***	***B***
consecutive digits	0	8
Consonant count	16	18
Vowel count	7	13
Digit count	0	8
Repeated letters	5	13

Five of our expanded linguistic traits are shown in Table 1. Taking two examples, A="/starnet/addons/slideshow_full.-php? album_name=289050446" and B="/tests/numbertotexttest.php," somewhere A has 9 successive digits and B has 0, the malicious payload is more likely to cause harm. Similar to counting the number of consecutive numbers, we may count the number of consecutive consonants. The amount of duplicate letters in a payload is equivalent to the entire amount of era the same letter appears in the data. This property is calculated to be 12 in case A and 4 in case B, with h, u, d, o, l, p, m, s, t, a, n, e, and s, p, t, e, respectively, serving as the repeating letters. Additionally, in cases A and B, the number of vowels in a payload is determined to be 12 and 6, respectively.

4. Evaluation

This paper describes a novel technique known as stacked autoencoder based Convolutional Neural Network (SA-CNN), which aims to offer adaptive and well-organized pack inspection. The first step in the untimely discovery prescription that SA-CNN provides is to scan the IP address of each new flow via SA-CNN.

4.1. Dataset

Table 2 lists the three datasets that were used in this study, two of which were unencrypted and one of which was. During the offline training phase, we selected labeled datasets from Github that included over forty thousand benevolent payloads and over five thousand malicious payloads in arrange to produce a trustworthy machine learning model.

Table 2. Data description

Dataset	*Samples*		*Class*	*Total*
HTTP CSTC 2010	Normal	4600	Uncrypted	70000
	Anomalous	25000		
Github paylods	Normal	4000	Uncrypted	45000
	Anomalous	6000		
CTU-BOTNET	Normal	10000	Encrypted	30000
	Anomalous	2000		

This was done to generate a reliable machine learning model without overfitting the dataset. The HTTP Common Security Information Collection (CSIC) 2010 is a collection of synthetic online requests and the associated web risks including SQL buffer overflow, injection, cross-site scripting, and so on. After the data is cleaned up, we choose a sample size for testing consisting of 36,000 typically delivered parcels and 24,000 outliers. This encrypted dataset includes both regular network traffic and botnet activity on a college campus's computer system. After the data was cleaned and prepared, we settled on a training set size of 20,000 regular flows and 20,000 botnet flows. A total of 1,000 "real world" flows and 1,000 "botnet-generated" flows were employed for testing. This was done to ensure there is no overlap between the two datasets used for training and testing. This encrypted dataset was collected from a university network and includes both typical network traffic and traffic from a botnet. After the data was cleaned and prepared, we settled on a training set size of 20,000 regular flows and 20,000 botnet flows. We selected 1,000 natural flows and thousand botnet flows as our test sets. This was done to ensure there is no overlap between the two datasets used for training and testing.

4.2 Feature extraction using Autoencoder stacking

The contribution layer, concealed layer, and production layer of an autoencoder, a kind of unsupervised learning structure, are all represented in Fig 2. Both an encoder and a decoder are required for the training of an autoencoder. The term "decoder" is used to describe the process of recovering the original data from the encoded form. Considering the unlabeled input dataset $\{x_n\}N_{n=1}$, where$x_n \in R^{m\times1}$, h_n stands for the concealed encoder vector derived from X_n , while $\hat{X}_n$ is the decoder vector of the production layer. Consequently, the encoding procedure is as follow:

$$g_m = e\left(u_{1v_m} + a_1\right) \quad (8)$$

When the encoder's weight matrix u_{1v_m}, the bias vector a_1, and the encoding function e are all present. Decoder process:

$$\hat{y}_m = h(u_2 g_m + a_2) \quad (9)$$

Where$h$, u_2, and $a_2$are the decoding function, weight matrix, and bias vector, respectively.

Reconstruction error is minimized by optimizing autoencoder parameter sets:

$$\emptyset(\Theta) = arg\min_{\theta,\theta'} \frac{1}{m}\sum_{j=1}^{m} k(v^j \hat{v}^j) \quad (10)$$

Where k represents a loss function.

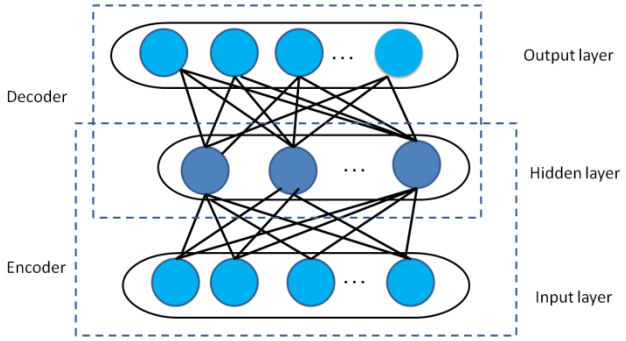

Fig.2. Structure of autoencoder

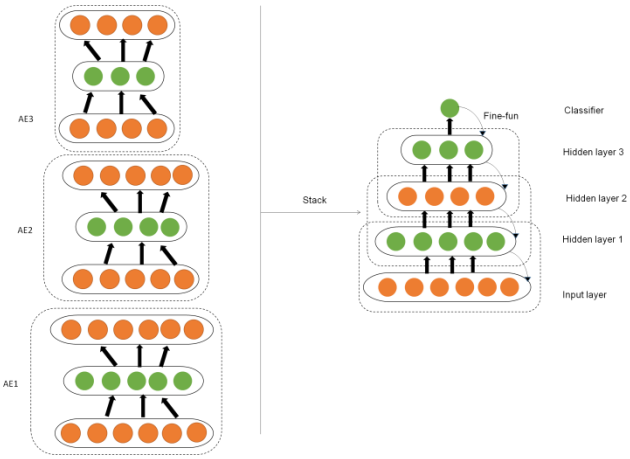

Fig.3. Structure of stacked autoencoders

Fig 3 depicts the architecture of SAEs, which entails layer-by-layer learning (unsupervised) to stack n autoencoders into n hidden layers, followed by supervised fine-tuning. Therefore, there are three stages to the SAEs-based approach:

- Train the first autoencoder and get the feature vector.
- The previous layer's feature vector is utilized as participation for the following deposit until training is complete.

- Back propagation algorithm (BP) minimizes the cost function and updates the weights with label guidance set to accomplish fine alteration once all concealed layers are learned.

4.3 Offline training test

The SA-CNN trains two binary classifiers, one for detecting unencrypted packets and the other for identifying encrypted packets.

4.3.1 Performance metrics

While assessing binary classification methods, performance metrics. Given a series of testing packets, the negative class designates the benign, while the positive class signifies malicious packets. Choose a few common categorization metrics to assess the effectiveness given a typical confusion matrix.

a. Calculate these evaluation metrics:

It's crucial to evaluate a model's performance on both training and testing data. The model's better accuracy in detecting on testing data suggests that it has learnt to generalize and identify malicious traffic in new circumstances.

$$TPR = \frac{TP}{TP+FN} \quad (11)$$

The True Positive Rate (TPR), also known as Recall, describes the proportion of malicious packets along with all harmful packets that have been recovered.

$$FPR = \frac{FP}{FP+TN} \quad (12)$$

The False Positive Rate (FPR) displays the proportion of benevolent packets that were mistakenly identified as malicious packets out of all the benevolent packets that were recovered.

Precision refers to the percentage of authentically malicious packets between all the harmful packets provided by the classified, and it indicates the numeral of authentically malicious packets included in the classed positive consequences.

$$Precision = \frac{TP}{TP+FN} \quad (13)$$

An ideal binary classifier is expected to have high Precision and Recall values, but in practice, they might be incongruent.

$$F_1 - score = \frac{2PR}{P+R} \quad (14)$$

Adopt the harmonic average of the two numbers to be the F1 score. F1 -score is a good way to show how well our classifier performs while taking into accounts both Precision and Recall.

4.3.2 Traffic training without encryption

The classifier for logistic deterioration is trained using 5-fold cross support. The dataset has an unbalanced distribution of data; this study uses the stratify approach throughout the validation phase. Additionally, logistic regression is contrasted with other widely used classifiers use a Receiver Operating Characteristic (ROC) curve, with false positive rate (FPR) on the horizontal axis and true positive rate (TPR) on the vertical axis, to evaluate the efficacy of binary classifiers. The ROC curve for six classifiers is shown in Fig 4.

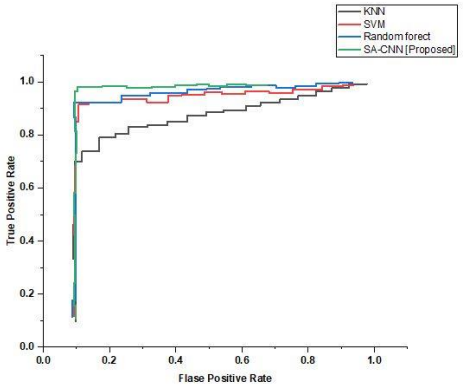

Fig.4. Classifier receiver operating characteristic curves

'The likelihood that a randomly picked compassionate payload is rated higher than an accidentally selected malicious payload is represented by the area under the ROC curve, also known as AUC (Area Under Curve). The precision-recall curve of various classifiers, which is a graph with recall on the x-axis and precision on the y-axis, may reveal a higher degree of efficiency'. When compared to other classifiers, SA-CNN and logistic regression have a more favorable precision-recall curve. Fig 5 & 6 and Table 3 displays the outcomes of training for several performance measures. High accuracy in recognizing malicious payloads is achieved by the SA-CNN and decision tree, although their training time is much higher than that of logistic regression.

Table 3. Classifier comparisons

Different classified	***Matrix***			
	Accuracy	**precision**	**recall**	**F1-score**
KNN	96	93	86	89
SVM	97	921	89	83
Random forest	94	94	93	94
SA-CNN	98	96	97	96

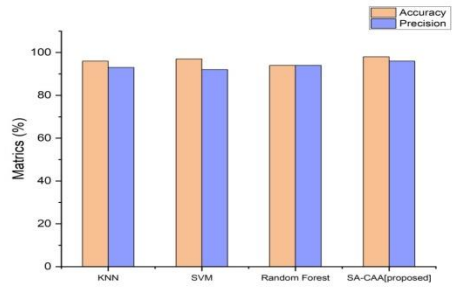

Fig.5. Accuracy and precision are contrasted

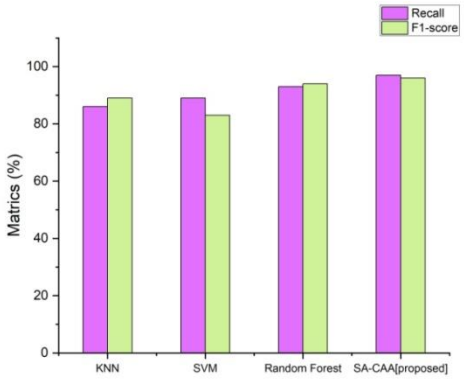

Fig.6. Comparing Recall and F1-score

The training time of SA-CNN surpasses 30 seconds, as shown in Fig 7, which is not ideal for models that are updated routinely. Logistic regression, on the other hand, has an average training time of 2.93 seconds and a 98.96% accuracy rate.

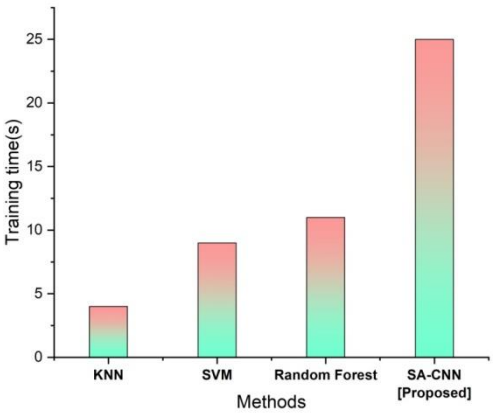

Fig.7. The training time of different classifiers

In conclusion, logistic regression outperforms other classifiers in identifying out-of-the-ordinary payloads. Using fivefold cross validation and fifty iterations, we get the learning curves for the logistic regression classifier by randomly picking 20% of the validation set. In fig. 8, which is below. It is clear that the training score starts out higher than 0.98 and increases consistently during the training time.

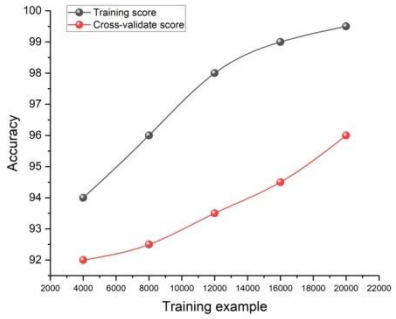

Fig.8. The logistic regression classifier

The validation score improves when more training instances are used, and the average score is higher than 0.925; nevertheless, it places a significant amount of emphasis on the use of conventional feature engineering. Uses deep learning models to give an end-to-end payload categorization lacking feature extraction. To encrypt unprocessed payloads, they use a word embedding technique. Next, they use a classification method based on Random Forest and RNN to extract characteristics from the unprocessed payloads. Although there is currently insufficient data to accurately assess SA-CNN controller performance, semi-supervised learning offers hope for improving deep packet inspection in SA-CNN.

4.3.3 Detection with new dataset

To provide preliminary IP address filtering, the SA-CNN controller makes use of a well-known IP blacklist from Cisco. The consequences of the linear prediction model detection are shown in Fig 9. Given that the present traffic pattern has minimal correlation with a large number of prior samples, each packet window is adaptively chosen based on the past ten samples (i.e., let N = 10).

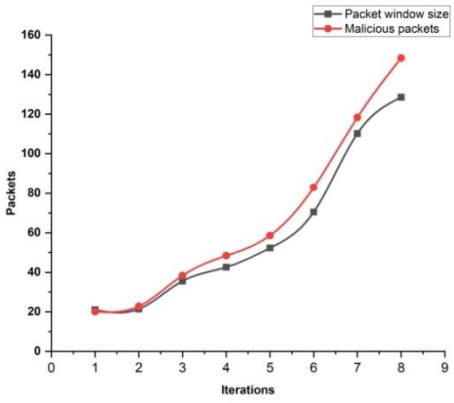

Fig .9. Adaptive packet sampling results based LP

The green one controls the packet window size, while the blue one handles any harmful packets that are uncovered. Packet window size is shown to have a linear connection with the number of malicious packets. When the packet

window size varies towards its maximum value, the greatest detection results are frequently achieved with a larger number of malicious packets; however, this comes at the expense of greater resource consumption.

5. Conclusion

In the existence of a huge amount of data, DPI in SDN is still limited. Although there are third-party DPI solutions, this study suggests a revolutionary stacked autoencoder based Convolutional Neural Network (SA-CNN) technique in SDN utilizing machine learning algorithms. Accuracy, precision, recall, and F1-score of the proposed SA-CNN are all very high at 98%, 96%, and 97% respectively. By separately training two binary classifiers, SA-CNN allows deep packet inspection of both clear text and encrypted data. Additionally, SA-CNN may sample potentially malicious packets using a linear prediction-based packet window. On the Ryu SDN controller and the Mininet platform, To evaluate SA-CNN using real-world datasets. With just modest overhead, SA-CNN can distinguish between encrypted and unencrypted transmission with a high degree of accuracy.

References

[1] Elsayed, M. S., Le-Khac, N. A., Dev, S., & Jurcut, A. D. (2020, August). Ddosnet: A deep-learning model for detecting network attacks. In 2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM) (pp. 391-396). IEEE.

[2] Chang, L. H., Lee, T. H., Chu, H. C., & Su, C. W. (2020). Application-based online traffic classification with deep learning models on sdn networks. Adv. Technol. Innov, 5(4), 216-229.

[3] Fouladi, R. F., Ermiş, O., & Anarim, E. (2022). A DDoS attack detection and countermeasure scheme based on DWT and auto-encoder neural network for SDN. Computer Networks, 214, 109140.

[4] Ujjan, R. M. A., Pervez, Z., Dahal, K., Khan, W. A., Khattak, A. M., & Hayat, B. (2021). Entropy based features distribution for anti-ddos model in sdn. Sustainability, 13(3), 1522.

[5] Lee, T. H., Chang, L. H., & Syu, C. W. (2020, June). Deep learning enabled intrusion detection and prevention system over SDN networks. In 2020 IEEE International Conference on Communications Workshops (ICC Workshops) (pp. 1-6). IEEE.

[6] Ahuja, N., Singal, G., & Mukhopadhyay, D. (2021, January). DLSDN: Deep learning for DDOS attack detection in software defined networking. In 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 683-688). IEEE.

[7] Yaser, A. L., Mousa, H. M., & Hussein, M. (2022). Improved DDoS Detection Utilizing Deep Neural Networks and Feedforward Neural Networks as Autoencoder. Future Internet, 14(8), 240.

[8] Garg, S., Kaur, K., Kumar, N., & Rodrigues, J. J. (2019). Hybrid deep-learning-based anomaly detection scheme for suspicious flow detection in SDN: A social multimedia perspective. IEEE Transactions on Multimedia, 21(3), 566-578.

[9] Ujjan, R. M. A., Pervez, Z., Dahal, K., Bashir, A. K., Mumtaz, R., & González, J. (2020). Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. Future Generation Computer Systems, 111, 763-779.

[10] Sindian, S., & Samer, S. (2020). An enhanced deep autoencoder-based approach for DDoS attack detection. Wseas Trans. Syst. Control, 15, 716-725.

[11] Qin, Y., Wei, J., & Yang, W. (2019, September). Deep learning based anomaly detection scheme in software-defined networking. In 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS) (pp. 1-4). IEEE.

[12] Yang, L., Song, Y., Gao, S., Xiao, B., & Hu, A. (2020, December). Griffin: an ensemble of autoencoders for anomaly traffic detection in SDN. In GLOBECOM 2020-2020 IEEE Global Communications Conference (pp. 1-6). IEEE.

[13] Novaes, M. P., Carvalho, L. F., Lloret, J., & Proença Jr, M. L. (2021). Adversarial Deep Learning approach detection and defense against DDoS attacks in SDN environments. Future Generation Computer Systems, 125, 156-167.

[14] Hashemi, M. J., & Keller, E. (2020, November). Enhancing robustness against adversarial examples in network intrusion detection systems. In 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN) (pp. 37-43). IEEE.

[15] Alzahrani, A. O., & Alenazi, M. J. (2021). Designing a network intrusion detection system based on machine learning for software defined networks. Future Internet, 13(5), 111.

[16] Al-jammaz, R. A. ., Rawash, U. A. ., Kashef , N. M. ., & Ibrahim, E. M. . (2023). A Framework for Providing Augmented Reality as a Service Provided by Cloud Computing for E-Learning. International Journal on Recent and Innovation Trends in Computing and Communication, 11(2s), 20–31. https://doi.org/10.17762/ijritcc.v11i2s.6025

[17] Morzelona, R. (2021). Human Visual System Quality Assessment in The Images Using the IQA Model Integrated with Automated Machine Learning Model . Machine Learning Applications in Engineering Education and Management, 1(1), 13–18. Retrieved from http://yashikajournals.com/index.php/mlaeem/article/view/5