Detecting Web Attacks Based on Clustering Algorithm and Multi-branch CNN

Pham Van Huong, Le Thi Hong Van, Pham Sy Nguyen

***Abstract*—This paper proposes and develops a web attack detection model that combines a clustering algorithm and a multi-branch convolutional neural network (CNN). The original feature set was clustered into clusters of similar features. Each cluster of similar features was generalized in a convolutional structure of a branch of the CNN. The component feature vectors are assembled into a synthetic feature vector and included in a fully connected layer for classification. Using K-fold cross-validation, the accuracy of the proposed method 98.8%, F1-score is 98.9% and the improvement rate of accuracy is 1.479%.**

***Tóm tắt*—Bài báo đề xuất và phát triển mô hình phát hiện tấn công Web dựa trên kết hợp thuật toán phân cụm và mạng nơ-ron tích chập (CNN) đa nhánh. Tập đặc trưng ban đầu được phân cụm thành các nhóm đặc trưng tương ứng. Mỗi nhóm đặc trưng được khái quát hóa trong một nhánh của mạng CNN đa nhánh để tạo thành một vector đặc trưng thành phần. Các vector đặc trưng thành phần được ghép lại thành một vector đặc trưng tổng hợp và đưa vào lớp liên kết đầy đủ để phân lớp. Sử dụng phương pháp kiểm thử chéo trên mô hình đề xuất, độ chính xác đạt 98,8%, F1-score đạt 98,8% và tỉ lệ cải tiến độ chính xác là 1,479%.**

Keywords—web attack detection; convolutional neural network (CNN); deep learning; K-means; multi-branch CNN.

Từ khóa—phát hiện tấn công Web; mạng nơ-ron tích chập (CNN); học sâu; K-means; CNN đa nhánh.

I. INTRODUCTION

Along with the exponential growth in the number of websites worldwide, the forms of attacks on this type of network service are also increasingly diverse. According to the Internet Live Stats, in November 2020, there are more than 1.8 billion websites worldwide. The attack methods on the web are increasingly diverse, typically: XSS, HTTP Request Smuggling, DoS, SQL Injection, etc. At the same time, the world has also recorded a positive trend of website security globally. Specifically, the CyStack Attack Map system recorded 392,300 attacks on the website, decreased more than 20% compared to the same period last year. This is partly due to the fact that prevention and detection methods have been actively developed. These measures are aimed at minimizing the damage from attacks on websites, increasing the proactivity of coping as well as preventing specific prevention measures of each business or unit.

There are many typical web attack detection methods such as static analysis, anomaly detection, using IDS/IPS, using Honey Pot/Honey Net, machine learning, deep learning, etc. Machine learning and deep learning are focused on development and application in most fields, such as image recognition, video recognition, medicine, entertainment, malware classification, etc. Web attack detection methods based on machine learning and deep learning have been applied vigorously and effectively since 2006 with a variety of attacks.

In deep learning algorithms, CNN shows the highest efficiency in classifying problems. Therefore, the CNN architectural models have been studied continuously for about 10 years. Since 2017, multi-branch CNN architecture was launched and applied effectively to a number of classification problems such as JPEG image classification, lesion identification in medicine, etc. Therefore, this paper proposes a method of detecting a web attack that uses a combination of DBSCAN clustering algorithm and multi-branch CNN.

The rest of the paper is organized as follows: *Section II* – Survey, analysis, synthesis of related research; *Section III* – Presentation on the basic idea, process and content of method's development; *Section IV* – Using K-means algorithm to cluster a feature set; *Section V* –

This manuscript is received on December 4, 2020. It is commented on December 22, 2020 and is accepted on December 22, 2020 by the first reviewer. It is commented on December 22, 2020 and is accepted on December 22, 2020 by the second reviewer.

Evaluation method; *Section VI* – Presenting our experiment; *Section VII* – Conclusion and trends of development.

II. Related works

There have been many research results using machine learning models in web attack detection problems with accuracy from 92% to over 99%. Most of the machine learning algorithms are used and compared to each other. In phishing attack detection problem, Babagoli, Aghababa, and Solouk (2018) used SVM algorithm to achieve 94.13% accuracy. Random Forest algorithm with only NLP-based features gives the best performance with the 97.98% accuracy rate for detection of phishing URLs [1]. In [2], the authors use most of machine learning algorithms to experiment with phishing detection using hyperlink information and the results show that Logistic Regression algorithm has the highest accuracy (98.42%). In SQL Injection attack detection, the authors used Naïve Bayes algorithm reached 93.3% [3]. In DoS, DDoS attack detection, the authors [4] uses an SVM algorithm based on web log traces.

Deep learning is known as a subset of machine learning, with outstanding performance in classification problems. Common deep learning models have also been used to detect several types of web attacks with great efficiency. Feng et al. (2018) proposed a novel neural network based on a classification method for detection of phishing web pages using a Monte Carlo algorithm and risk minimization principle. The CNN model [5] is used to detect website anomalies based on HTTP requests. The Stacked Auto Encoder (SAE) model [6] is applied for anomaly detection in web application firewall. Some other results such as: DoS attack detection based on Restricted Boltzmann Machine [7], detection of code injection attacks on hybrid applications using Hybrid Deep Learning Network (HDLN) between CNN and LSTM achieves accuracy of over 97.5% [8], etc.

In addition, there are some studies using a combination of machine learning/deep learning algorithms to classify attacks on websites with quite good results. An example would be combining the neural network approach with reinforcement learning for phishing attack classification (Smadi, Aslam, and Zhang - 2018). However, most of the above research results focus on detecting and warning about one or a few specific types of attacks on the websites, yet to detect diverse types of attacks.

Associative rule mining and clustering techniques using Apriori, FP-Growth or K-means algorithms are not too new in the field of big data mining [9]-[11]. K-means was widely applied and integrated in many clustering tools such as ELKI, WEKA, etc. Recently, this clustering algorithm is still receiving growing attention in terms of parameter selection for meaningful research results and good performance [12], [13].

In 2017, multi-branch CNN was proposed by Amerini et al to detect double JPEG image compression. It is then further developed in the direction of proposing another feature set for relatively high accuracy (average between 95% - 99%) [14]. In 2019, the research groups continued to propose branching CNN architecture for multiple sclerosis lesion segmentation [15], or for myocardial infarction screening from ECG images [16]. Therefore, it is used effectively in medicine. There are very few research results that use this architecture for the web attack detection problem [5].

Based on the above survey results, this paper proposes new methods to Web attack detection based on the combination of K-means clustering algorithm and Multi-branch CNN. Our method will be developed, experimented and evaluated in the following sections.

III. Idea and the mathematical model

A. Basic idea

The key idea of our paper is to use clustering algorithms to split an original feature set into the subsets corresponding to clusters; and put them to branches of a CNN to classify. Each cluster is put into a branch to generalize features to create a component feature vector. The component feature vectors are joined to generate a synthetic feature vector. This vector is put into a fully connected layer of CNN to classify. Because the features in a cluster have the closest metrics, it is more efficient to build the component feature vector for each cluster.

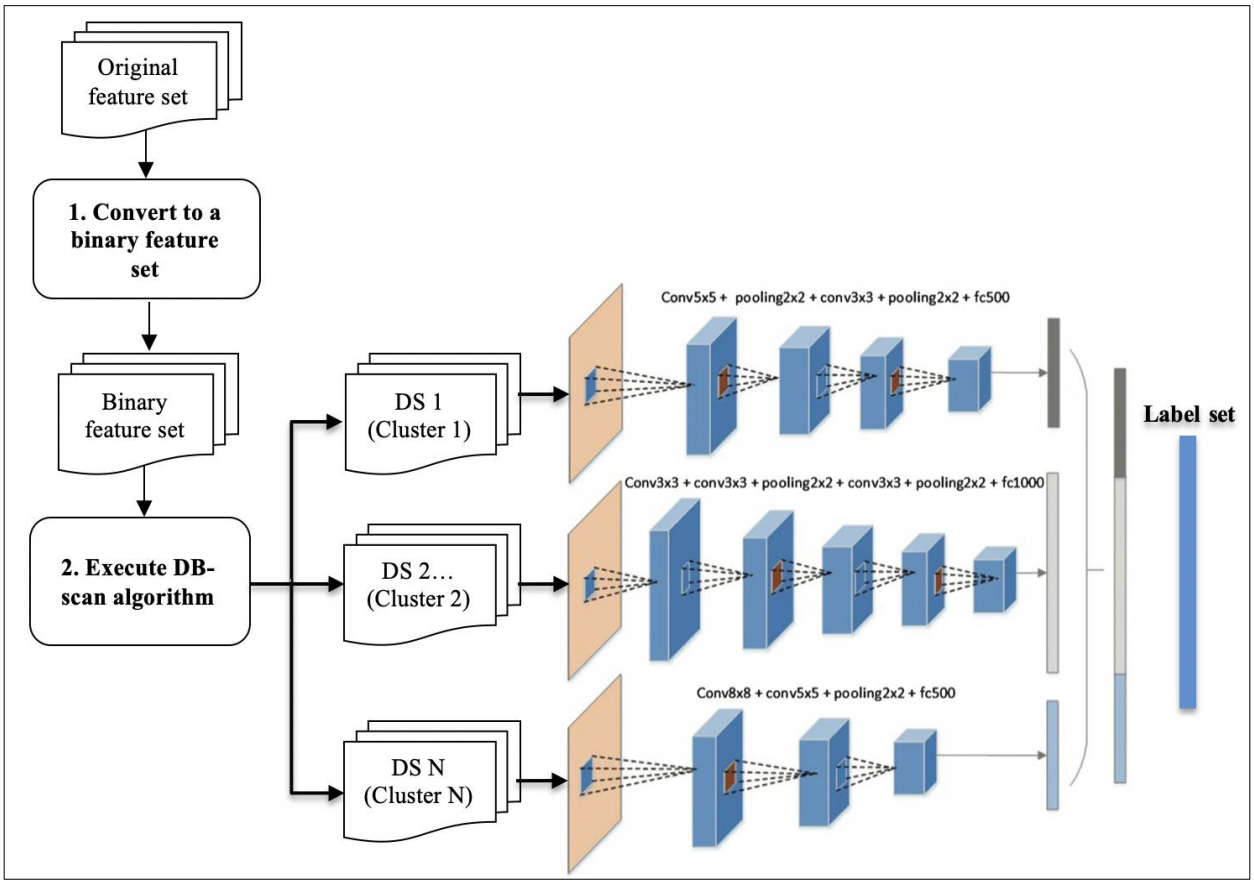

Fig. 1. Overall research model.

B. Building the mathematical model of the problem

Definition 1 – Component feature vector

A component feature vector is the feature vector generated by a branch of a CNN, is described by Equation (3).

Definition 2 – Synthetic feature vector

A synthetic feature vector is the feature vector created by joining component feature vectors described by Equation (4).

As shown in Fig. 1, the original feature set D is clustered by K-means algorithm to K clusters shown in Equation (2). And, the overall mathematical model of the problem is described by Equations (1) to (5).

$$f: D \to O \tag{1}$$

$$D = \bigcup_{i=1}^{K} D_i \tag{2}$$

$$v_i = f_{CNN}^{i}(D_i) \tag{3}$$

$$v = \bigcup_{i=1}^{K} v_i \tag{4}$$

$$f': V \to O \text{ and } V = \{v\} \tag{5}$$

The features in each cluster have similarities, so when using convolution and filtering part of a CNN branch, we obtain better generalization features. At the same time, each component feature vector is generated on a CNN branch so it also carries the characteristics of each cluster. Each component feature vector is called v_i. The synthetic vector v is formed by combining component features v_i.

Based on the overall model of the problem, the steps of building, analyzing, testing and evaluating methods will be presented in detail in the following sections.

IV. FEATURE SET CLUSTERING BASED ON K-MEANS ALGORITHM

K-means is one of the most popular clustering algorithms. K-means clustering algorithm computes the centroids and iterates until it finds

optimal centroid. It assumes that the number of clusters is already known. In this paper, we use K-means algorithm to cluster the original feature set to K subsets of features. K-means algorithm is described as follows.

K-means algorithm:

Input:

- A set of features.
- Number of clusters K.

Output: K subsets of features

Algorithm:

1 Initialize k cluster centroids randomly

$$\mathbb{C}^{(0)} = \{m_1^{(0)}, m_2^{(0)}, \ldots, m_k^{(0)}\} \quad (6)$$

2 Put each point into the cluster which has the nearest centroid

$$\mathbb{S}_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\|^2 \le \|x_p - m_j^{(t)}\|^2\} \ , \forall j, 1 \le j \le k \quad (7)$$

Stop if clusters do not change from the previous step

3 Update centroids

$$m_i^{(t+1)} = \frac{1}{|\mathbb{S}_i^{(t)}|} \sum_{x_j \in \mathbb{S}_i^{(t)}} x_j \quad (8)$$

V. EVALUATING THE METHOD

In order to evaluate the proposed method, we used a *K-fold* cross-validation method and measures such as *Accuracy*, *Precision*, *Recall* and *F1-score*. These measurements are calculated using Equation (9), (10) and (11).

$$Precision = \frac{TP}{TP + FP} \quad (9)$$

$$Recall = \frac{TP}{TP + FN} \quad (10)$$

$$F_1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (11)$$

where,

- *TP* is the true number of classified patterns of attack state.
- *FP* is the false number of classified patterns of attack state.
- *TN* is the true number of classified patterns of normal state.
- *FN* is the false number of classified patterns of normal state.

VI. EXPERIMENT

A. Experimental model

To evaluate the proposed method, we conducted experiments as shown in Fig. 2. In

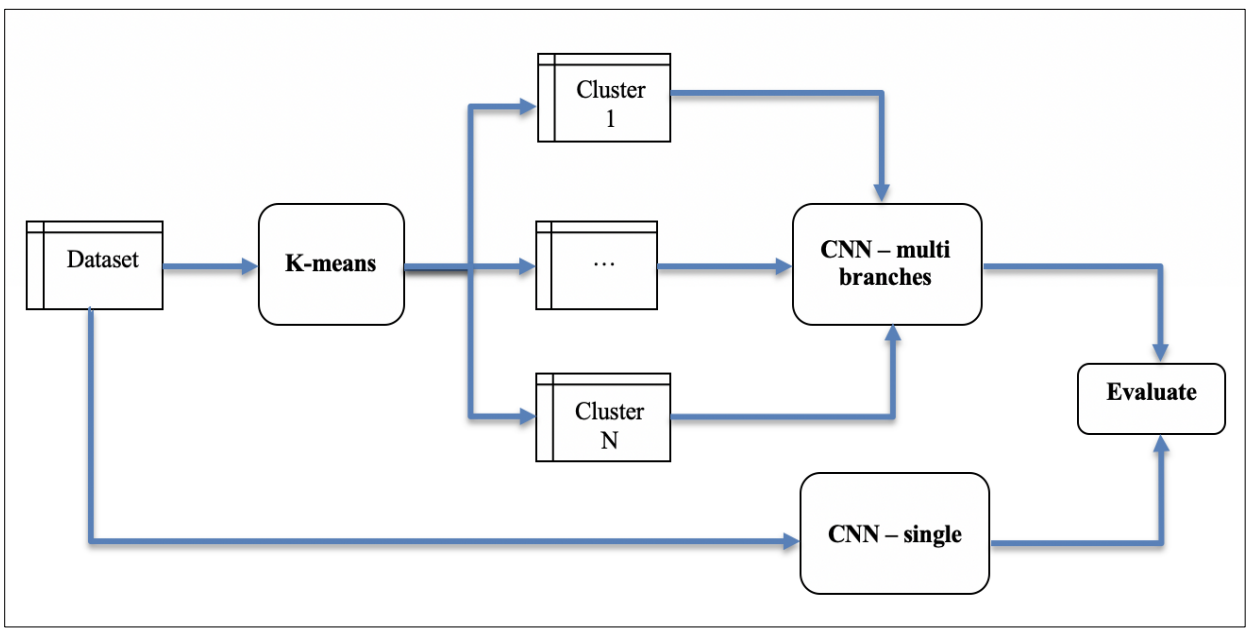

Fig. 2. Experimental model.

experiments, an original feature set is clustered into three clusters; the original feature set is passed through a one-branch CNN and each cluster is passed through a branch of a multi-branch CNN.

B. Experimental program and data

In this experiment, we installed the web attack detection program according to CNN in Python language, using the TensorFlow library. The two CNN network structures installed in the program consist of a one-branch CNN and a multi-branch CNN, described in Fig. 4 and Fig. 3. The multi-branches have three branches corresponding to the three clusters, with 585, 835 and 223 elements. To do our experiment, we use the dataset in [17].

```
# CNN bracnch for Cluster 1
#mô hình:conv 5x5+pool2x2 => conv3x3+pool2x2=> FC500
input_cluster1 = Input(shape=(CLUSTER1_SIZE,CLUSTER1_SIZE,1))
conv11 = Conv2D(32, kernel_size=5, activation='relu', padding="same"
                ,input_shape=(CLUSTER1_SIZE, CLUSTER1_SIZE, 1))(input_cluster1)
pool11 = MaxPooling2D(pool_size=(2, 2))(conv11)

conv12 = Conv2D(32, kernel_size=3, activation='relu', padding="same")(pool11)
pool12 = MaxPooling2D(pool_size=(2, 2))(conv12)

flatten_cluster1 = Flatten()(pool12)
hidden_cluster1 = Dense(1024, activation='relu')(flatten_cluster1)

# CNN bracnch for Cluster 2
#conv3x3=> conv3x3+pool2x2=> conv3x3+pool2x2=> FC1000
input_cluster2 = Input(shape=(CLUSTER2_SIZE,CLUSTER2_SIZE,1))
conv21 = Conv2D(32, kernel_size=3, activation='relu', padding="same"
                ,input_shape=(CLUSTER2_SIZE, CLUSTER2_SIZE, 1))(input_cluster2)
#pool21 = MaxPooling2D(pool_size=(2, 2))(conv21)

conv22 = Conv2D(32, kernel_size=3, activation='relu', padding="same")(conv21)
pool22 = MaxPooling2D(pool_size=(2, 2))(conv22)

conv23 = Conv2D(32, kernel_size=3, activation='relu', padding="same")(pool22)
pool23 = MaxPooling2D(pool_size=(2, 2))(conv23)

flatten_cluster2 = Flatten()(pool23)
hidden_cluster2 = Dense(1024, activation='relu')(flatten_cluster2)

# CNN bracnch for Cluster 3
#conv8x8=>conv5x5+pool2x2=>FC500
input_cluster3 = Input(shape=(CLUSTER3_SIZE,CLUSTER3_SIZE,1))
conv31 = Conv2D(32, kernel_size=8, activation='relu', padding="same",
                input_shape=(CLUSTER3_SIZE, CLUSTER3_SIZE, 1))(input_cluster3)
#pool31 = MaxPooling2D(pool_size=(2, 2))(conv31)

conv32 = Conv2D(32, kernel_size=5, activation='relu', padding="same")(conv31)
pool32 = MaxPooling2D(pool_size=(2, 2))(conv32)

flatten_cluster3 = Flatten()(pool32)
hidden_cluster3 = Dense(1024, activation='relu')(flatten_cluster3)
```

Fig. 3. Experimental Structure of CNN-multi-branches.

```
tf.reset_default_graph()

network = input_data(shape=[None, IMG_SIZE, IMG_SIZE, 1])

network = conv_2d(network, 32, 2, activation='relu')
network = max_pool_2d(network, 6) #3

network = conv_2d(network, 24, 1, activation='relu')
network = max_pool_2d(network, 3) #3

network = fully_connected(network, 576, activation='relu')
network = dropout(network, 0.6) #5

network = fully_connected(network, N_CLASSES, activation='softmax')
network = regression(network)
```

Fig. 4. Experimental structure of a CNN-1branch.

C. Feature conversion

In order to create binary matrices inputted to a CNN, we convert the original feature set to a binary feature set as shown in Fig. 5. and Fig. 6. Fig. 5 shows a part of the query string, used as a raw feature, having Xpath and XSS labels. Fig. 6 shows some binary features converted by raw features.

password=FrAmE30.&provincia=24&login=%3C%21--&dni=56&direccion=Bonsai+Street+123&apellidos=Smith&ciudad=Buenos+Aires&nombre=John&ntc=56&cp=56&email=w3af%40email.com&modo=insertar	XPath
modo=insertar&login=c2&password=c2&nombre=carlos&apellidos=perez&email=perez@yahoo.com&dni=23453457Z&provincia=1&cp=12354&B1=Confirmar&ciudad=Madrid&ntc=1234567890123456&direccion=%35%31%2c%35%32%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%34%2c%35%30%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%37%2c%35%36%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%32%2c%35%37%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%32%2c%35%37%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%33%2c%34%38%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%33%2c%35%34%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%33%2c%34%39%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%33%2c%35%33%2c%33%37%2c%35%30%2c%35%33%2c%35%30%2c%36%37%2c%35%37%2c%35%36%	XSS

Fig. 5. A part of query string in the original feature set.

F1639	F1640	F1641	F1642	F1643	Label
0	0	0	0	1	0
0	0	0	0	1	4
0	0	0	0	1	0
0	0	0	0	1	4
0	0	0	0	1	1
0	0	0	0	1	1
0	0	0	0	1	2
0	0	0	0	1	3
0	0	0	0	1	4
0	0	0	0	1	1
0	0	0	0	1	2
0	0	0	0	1	2
0	0	0	0	1	3

Fig. 6. A part of CNN feature set.

D. Experimental results and evaluation

The accuracy and relevant measurements when experimenting on the three data sets with CNN model by the cross-testing method are summarized in Table 1. The average improvement rate is 1.479%. Comparing the improvement level of the proposed method when experimenting on 3 clusters, it is summarized in chart form as Fig. 7.

As shown in Table 2, compared with some machine learning models in the study [18], including SVM, PCA, etc., the proposed model has higher accuracy. At the same time, the use of the K-means algorithm to group the features also improves the accuracy. This is because after clustering, we obtain groups of similar features, so the generalization of features in the convolution layers is more efficient.

TABLE 1. EXPERIMENTAL RESULTS

Models	Times										Average	
	1		2		3		4		5			
	F1-Score	Acc	F1-Score	Acc	F1-Score	Acc	F1-Score	Acc	F1-Score	Acc	F1-Score	Acc
CNN-1branch	0.962	0.967	0.974	0.965	0.968	0.983	0.975	0.981	0.969	0.973	0.970	0.974
CNN-multi-branches	0.985	0.986	0.989	0.991	0.983	0.984	0.991	0.995	0.995	0.985	0.989	0.988
Improvement rate (%)	2.391	1.965	1.540	2.694	1.550	0.102	1.641	1.427	2.683	1.233	1.960	1.479

TABLE 2. COMPARING TO OTHER METHODS

Method	Naive bayes	AGGRE GATE_ANY	Auto encoder	PCA	CNN
Acc.	0.941	0.933	0.906	0.737	0.988

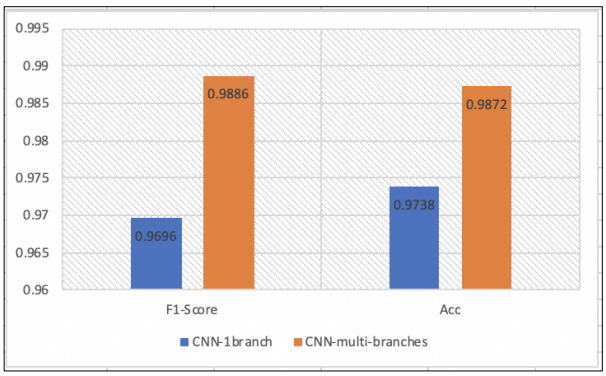

Fig. 7. Comparison of CNN-1 branch and CNN-multi-branches

VII. CONCLUSION

The main contribution of this paper is to propose and develop the new method of web attack detections, associated clustering by K-means algorithm and classifying by a multi-branch CNN. The proposed method is evaluated using K-fold cross-validation with good results. Our method is better than the original method on both F1-score and accuracy.

Despite the positive results, this paper still has some limitations such as: the number of classes is small, the number of samples is limited, and the cluster number is fixed. Therefore, we will continue to research and improve the methodology in the paper including: experimenting with other machine learning/deep learning models; studying on dynamic cluster numbers; experimenting with other actual data sets with a higher number of classes and more diverse forms of attacks.

REFERENCES

[1] Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, Banu Diri, Machine learning based phishing detection from URLs, Expert Systems With Applications 117, 2019, pp. 345–357.

[2] Ankit Kumar Jain1 · B. B. Gupta, A Machine Learning based Approach for phishing detection using hyperlinks information, © Springer-Verlag GmbH Germany, part of Springer Nature 2018.

[3] Anamika Joshi, Geetha V, SQL Injection Detection using Machine Learning, 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014.

[4] Yuchun Tang, Zhenyu Zhong, Yuanchen He, System and Method for Detection of DoS Attacks, Apr. 25, 2013.

[5] Ming Zhang, Boyi Xu, Shuai Bai, Shuaibing Lu, and Zhechao Lin, A Deep Learning Method to Detect Web Attacks Using a Specially Designed CNN, ICONIP 2017, Part V, LNCS 10638, 2017, pp. 828–836.

[6] Ali Moradi Vartouni, Saeed Sedighian Kashi, Mohammad Teshnehlab, An Anomaly Detection Method to Detect Web Attacks Using Stacked Auto-Encoder, 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), 2018.

[7] Ruibo Yan, Xi Xiao, Guangwu Hu, Sancheng Peng, Yong Jiang, New deep learning method to

detect code injection attacks on hybrid applications, The Journal of Systems and Software 137, 2018, pp. 67–77.

[8] Yadigar Imamverdiyev, Fargana Abdullayeva, Deep Learning Method for Denial of Service Attack Detection Based on Restricted Boltzmann Machine, Mary Ann Liebert, Inc., Big Data, Volume 6 Number 2, 2018.

[9] Coenen, F., Goulbourne, G. and Leng, P., Tree Structures for Mining association Rules, Journal of Data Mining and Knowledge Discovery, Vol 8, No 1, 2003, pp. 25-51.

[10] Asantha Thilina, Shakthi Attanayake, Sacith Samarakoon, Dahami Nawodya, Lakmal Rupasinghe, Nadith Pathirage, Tharindu Edirisinghe, Kesavan Krishnadeva, Intruder Detection Using Deep Learning and Association Rule Mining, IEEE International Conference on Computer and Information Technology, 2016.

[11] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, In Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (KDD), 1996, pp. 226–231.

[12] Junhao Gan, Yufei Tao, DBSCAN revisited: Mis-Claim, Un-fixability and Approximation, SIGMODE 2015.

[13] Erich Schubert, Jorg Sander, Martin Ester, Hans-Peter Kriegel, Xiaowei Xu, DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN, ACM Trans. Database Syst. 42, 3, Article 19, 2017.

[14] 14. Bin Li, Hu Luo, Haoxin Zhang, Shunquan Tan, Zhongzhou Ji, A multi-branch convolutional neural network for detecting double JPEG compression, Arxiv, 2017.

[15] Shahab Aslani, Michael Dayan, Loredana Storelli, Massimo Filippi, Vittorio Murino, Maria A Rocca, Diego Sona, Multi-branch Convolutional Neural Network for Multiple Sclerosis Lesion Segmentation, Arxiv, April 2019.

[16] Pengyi Hao, Xiang Gao, Zhihe Li, Jinglin Zhang, Fuli Wu, Cong Bai, Multi-branch fusion network for Myocardial infarction screening from 12-lead ECG images, Computer Methods and Programs in Biomedicine 184, 2020.

[17] Web attack detection dataset: https://github.com/DuckDuckBug/cnn_waf

[18] Pan Yao, Sun Fangzhou, Teng Zhongwei, White Jules, Schmidt Douglas, Staples Jacob and Krause Lee, Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications, 2019.

ABOUT THE AUTHOR

Pham Van Huong

Workplace: Academy of Cryptography Techniques

Email: huongpv@actvn.edu.vn

Education: Received Bachelor's degree in 2005, Master's degree in 2008 and PhD in 2015 in Information Technology from University of Engineering and Technology, VNU.

Recent research direction: IoT, AIoT, embedded software optimization and big data, deep learning for information security.

Le Thi Hong Van

Workplace: Academy of Cryptography Techniques

Email: lthvan@actvn.edu.vn

Education: Received Engineer's degree in 2009 and Master's degree in 2013 in Information Security from Academy of Cryptography Techniques.

Recent research direction: information security, cryptography, IoT and application of AI, machine learning for information security.

Pham Sy Nguyen

Workplace: Informatics center, The Government Office

Email: phamsynguyen@chinhphu.vn

Education: Received Engineer’s degree in Information Security in 2013; received Master’s degree in Information Security in 2016 from Academy of Cryptography Techniques.

Recent research direction: web hacking, malware detection, information security.