A hybrid of CNN and LSTM methods for securing web application against cross-site scripting attack

Raed Waheed Kadhim[1], Methaq Talib Gaata[2]
[1]Informatics Institute for Postgraduate Studies /Iraqi Commission for Computer and Informatics, Iraq
[1,2]Computer Science Department, University of Mustansiriyah, Baghdad, Iraq

Article Info

Article history:

Received Jul 6, 2020
Revised Sep 7, 2020
Accepted Oct 21, 2020

Keywords:

Algorithms
CBOW
CNN and LSTM
Confusion matrix
Word2vec
XSS

ABSTRACT

Cross-site scripting (XSS) is today one of the biggest threatthat could targeting the Web application. Based on study published by the open web applications security project (OWASP), XSS vulnerability has been present among the TOP 10 Web application vulnerabilities. Still, an important security-related issue remains how to effectively protect web applications from XSS attacks. In first part of this paper, a method for detecting XSS attack was proposed by combining convolutional neural network (CNN) with long short term memories (LSTM), Initially, pre-processing was applied to XSS Data Set by decoding, generalization and tokanization, and then word2vec was applied to convert words into word vectors in XSS payloads. And then we use the combination CNN with LSTM to train and test word vectors to produce a model that can be used in a web application. Based on the obtaned results, it is observed that the proposed model achevied an excellent result with accuracy of 99,4%.

Corresponding Author:

Raed Waheed Kadhim
Informatics Institute for Postgraduate Studies
Iraqi Commission for Computer and Informatics, Iraq
Email: raad742002@yahoo.com

1. INTRODUCTION

Most of the existing web applications are widely available and open to all people, even hackers, provided by companies that provide the web service [1-4]. A large portion of these attacks are XSS type, which has been considered one of the top threats in the OWASP classification for several years [5-8]. Hackers can, for example, access multiple resources, sensitive data, phishing attacks, misinformation, inserting malicious content, tampering Web pages, hijacking user sessions, controlling user'sbrowser and access a database when they succeed in hacking them into web applications [9-11].

XSS attacks are classified into three main categories: non-persistent XSS attacks, persistent XSS attacks, and DOM-based XSS attacks [12-17]. The XSS attack is easy to implement, but it is difficult to detect and prevent this attack because of different browsers used. Furthermore, Figure 1 shows the XSS attack architecture, as it is noted from this figure that the intruder adds an injected code to a website and uses this site as a prey to catch a victim, and the intruder can read the cookies of the website user or obtain the password of the user's login account, disguised as a user login website, to obtain permissions from the user [18-22].

Non-persistent XSS attacks impact the current user-visited page, allowing the user to access the attacker's tampered connection. When the user visits the page, the embedded script for the attack is executed by the browser of the User [23, 24]. Persistent XSS stored server-side data of the attacker and the attack can start with the attack data. XSS attacks that are not persistent and XSS attacks that are persistent can be divided into three categories: stored XSS attacks, reflective XSS attacks, or dom-related XSS attacks [25].

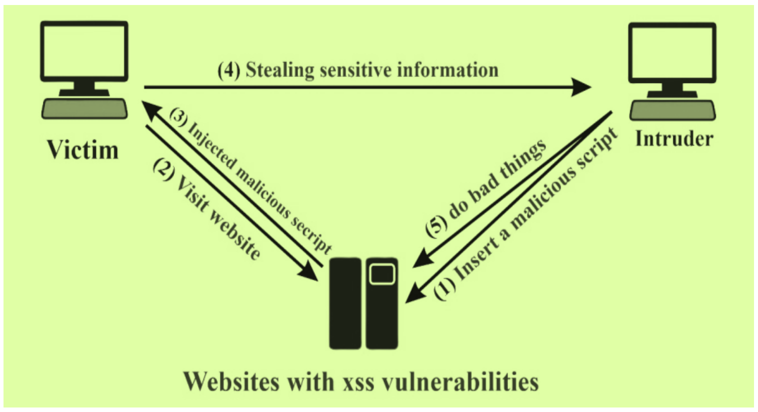
(4) Stealing sensitive information
Victim
Intruder
(3) Injected malicious secript
(2) Visit website
(5) do bad things
(1) Insert a malicious script
Websites with xss vulnerabilities

Figure 1. The architecture of XSS attack

A number of researchers have proposed a set of methods to detect XSS threats discovered by artificial intelligence in recent decades such as:

BA Vishnu et al. [4] used three MLalgorithms (SVM, Naive Bayes, J48 decision trees) to detect whether or not there are XSS threats in the URL and java script. S Rathore et al. [5] ten traditional algorithms of ML are used to detect whether or not the SNS websites contain XSS attacks. Likarishet al. [6] proposed a ML algorthims in web applications to detect the obscured malicious JavaScript, this method established 65 features and trained four clasification classifiers. Ahmed and Ali [7] Proposed use of genetic algorithms to detect XSS attacks, they tested their solution to web applications based on MySQL and PHP.

Wang and Zhou [8] Novel has proposed relying on new technologies. To detect XSS attacks, for example, HTML5 evaluated the performance of the proposed system using XSSer, a common tool created by OWASP [9]. Built an S2XS2 tool to detect server-side XSS attacks based on the automatic border injection system and the principle of policy generation, to test this process, they use four JSP programmes. Nathan et al. [10] proposed system to detect XXS attacks based on random forest as classifier. Vartouhi [11] Proposed system for analyzing HTTP traffic to extract vector features by using n-gram and classifying based on forest isolation (Forest) method.

Grosse Kathrin [12] proposed an adversarial example, Malware detection algorithm generation which allowed 63 percent of malware detectors designed to successfully bypass by neural Networks. Yao Wang, Wan-dong Cai, and Peng-cheng Wei, 2016 [13] Present a modern, deep learning system for detecting JavaScript malicious code. [14], 2018, ScriptNet system proposed to detect XSS attacks using static, dynamic analysis and deep recurrent neural classification. In this paper, the researcher presented a method for detecting XSS attacks by relying on mixing two methods of deep learning, namely the CNN method and the LSTM method based on XSS global dataset, and after obtaining a trained model, it is used for secure chat web application.

The contribution of this research includes three main contributions:

a) Feature extaraction based on extract information on semantics by using Word2vec method.
b) Mixing of the CNN method with the LSTM method to detect XSSattacks, which achieved an accurate 99,4%.

The rest of the paper is organized as follows. Research methods is presented in Section 2, and in Section 3 we give a detailed description of proposed system. In Section 4, we conduct the experiments and evaluation results. Finally, in Section 5, we summarize our work and discuss further work.

2. RESEARCH METHOD

Securing of Web application is critical issue, because of increasing cybercrimes. In this section, the research methods have been discussed.

2.1. Convolutional neural network (CNN)

Network attack detection has recently become more important to social networking data protection, as security threats such as cross-site scripting, click jacking, DDOS, probe and identity theft have increased. The aim of detecting network attacks is to differentiate between hostile activities and network traffic [16]. Machine learning (ML) has been successfully applied in different areas of computer science, such as image processing and computer vision. One of the popular ML techniques in the field of intrusion detection is the

K-Nearest Neighbor (K-NN) [17]. However, convolutions neural network (CNN) remains the commonly used deep learning model for large-scale image recognition and classification. The CNN works by directly extracting images. CNNs learn to detect various image features using tens or hundreds of hidden layers. Each hidden layer increases the complexity of the image characteristics learned. The working principle of the CNN relies on direct extractin of image features. CNNs are trained to detect various image features with varying numbers of hidden layers and each hidden layer adds to the complexity of the learned image features [18].

2.2. Longshort term memories (LSTM)

Long-form short memory (LSTM) networks are a cutting-edge sequence learning technique. They are less common for time series forecasts, but are intrinsically suitable for this domain. LSTM is an RNN variant capable of learning long-term dependencies. The Hochreiter and Schmidhuber first proposed and refined LSTMs by many other researchers. They work well on a wide range of problems and are the most used RNN type. The mathematical formulation of LSTM units prior to diving in the theory behind the concept shows how the hidden state ht-t time t is determined from the previous hidden state ht-1 [19]:

The Input gate is dysplaied as,

$$i(t) = \sigma(W(i)x(t) + u(i)h(t-1)) \quad (1)$$

Forget Gate calculated as,

$$f(t) = \sigma\ (W(f)x(t) + u\ (f)h(t-1)) \quad (2)$$

and, the Output/ Exposure gate as,

$$o(t) = \sigma\ (W(o)x(t) + u(o)h(t-1)) \quad (3)$$

and the New memory cell as,

$$c\tilde{}(t) = \tanh(W(c)x(t) + u(c)h(t-1)) \quad (4)$$

Lastly, the final memory cell is calculated as,

$$c(t) = f\ (t)o\ c\tilde{}\ (t-1) + i(t)o\ c\tilde{}(t) \quad (5)$$

And,

$$h(t) = o(t)o\tanh(c(t)) \quad (6)$$

2.3. Word2vec

Google released Word2vec in 2013 as a deep learning tool that comprises of the Continuous Bag of Words (CBOW) model &the Skip-Gram model. The role of the CBOW model is to perform context-based prediction of the current word while the surrounding words for the given current wordare predicted by the Skip-Gram model [20].

2.4. Evaluation methods

There are many types of evaluation methods that have been designed for performance evaluation of new methods. In this work, the most common types of evaluation methods which are are Accuracy, Recall and F1 Score. Moreover, there are four classes in the confusion matrix, these are the True Positiveswhich specifies amount of XSS samples that are correctly classified, the False Positiveswhich portrays the number of Non-XSS samples wrongly classified as XSS, the True Negative that indicate the number of correctly classified Non-XSS samples, and the False Negatives that capture the the number of XSS samples wrongly classified as Non-XSS. In this work, the calculation of accuracy is presented as in (7),

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (7)$$

wherein the precision was calculated as in (8),

$$Recall = \frac{T_p}{T_p + F_n} \quad (8)$$

Also, the sensitivity was calculated as in (9),

$$F - Measure = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \quad (9)$$

3. PROPOSED SYSTEM

Deep learning has progressed greatly in computer vision, the processing of natural languages and artificial intelligence, and it has also begun to develop in the field of security and move to practical applications. The experiments carried out using this proposed method mainly use text classification methods to detect XSS attacks by deep learning. Firstly, we process input data, including Decoding, generalization and tokaniztion Input data. Then, with a word2vec model, we extract features. Finally, these features have been placed into CNN-LSTM to train a classification to differentiate XSS from normal samples. Figure 2 shows the architecture of the proposed method. In the next subsection, the processing stepes of the proposed methods have been illustrated.

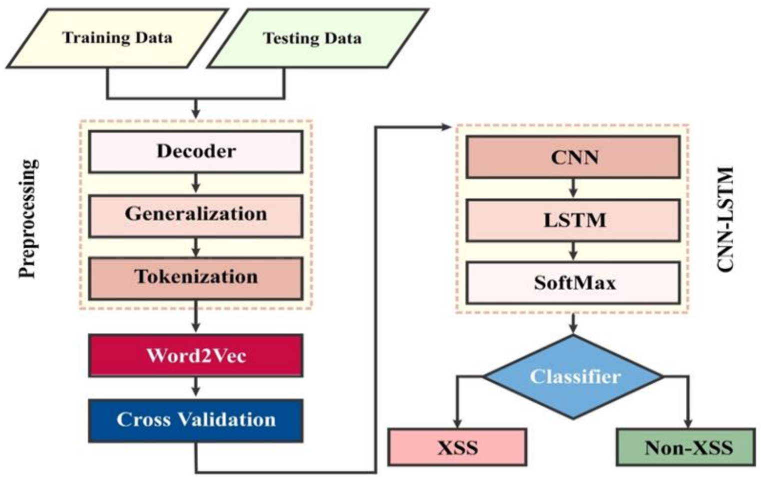

Figure 2. The architecture of the proposed method

3.1. Preprocessing stage

This is the first stage of the proposed methods after import data to the system, it is consistes of three steps which are decoded, generalized and tokenized, as explained below:

3.1.1. Decoder

Attackers may avoid conventional filtration or validation based on regular expressions by relying on encoding techniques such as Hex encoding, URL encoding, Unicode encoding, HTML entity encoding, etc. Hence, this study proposesadecoder for recurrent testing and conversion of all the dinput data ecoding options into the original form. Words consisting of characters and numbers Respectively, for the reduction of the number of participles the number of numbers and the hyperlinks must be simplified to replace numbers by "0" and the hyperlinks to http: / u. After the processing of the decoder, for example, an obscured XSS payload using HTML:encoding“<script>&#97; &#108; &#101; &#114; &#116; &#40; &#49; &#41;</script>” is restored to “<script>alert (1) </script>”.

3.1.2. Generalization

To reduce the interference of redundant and irrelevant information, the decoded data is generalised by the following steps: First, we replace several URLs of the input data using 'http:/website. The numbers in the data are then replaced by "0." And the string is replaced with "param string" as a function parameter. Furthermore, other special characteristics, such as control and blank characters were eliminated.

3.1.3. Tokenization

This study designed a set of regular expressions for input data customization based on the scripting language features. The Tokenization Classification is shown in Table 1.

Table 1. Classification of tokenization

Classification	List
Start Label	<script>, <frame>, <img, <body>, etc
End Label	</script>, </frame>, </body>, etc
Windows Event	onerror=, onload=, onblur=, oncut=, etc
Function Name	alert(, prompt(, String.fromCharCode(, etc
Script URL	javascript:,vbscript:, etc
Others	>,), #, etc

All step of preprocessing stage above are summarized in an algorithm (3.1) of as shown:

```
Algorithm 3.1: Preprocessing Stage Algorithm
Input      Read vector (V).
Output     Number of scripts.
Step1      Read each script until reach the end of file
Step2      Converts the input data into the original form( Decoder)
Step3      Replaced numbers in the input data with "0", and the string as a function
           parameter with "param_string". (Generalization).
Step4      Separate each word from the scripts to obtain a tokens (Tokenization).
Step5      Return scripts list (Data)
Step6      Return token list from previous step (Word)
Step7      Calculate the frequency for each token (Vocabulary)
           For each token in VocabularyList
           If token found in list
           Then
Step8      Take token
           Else
           replace it with (-1)
           End if
Step9      Return scripts list
```

3.2. Word2Vector

In the Word2Vec stage we 're going to have features extraction, we need to create a vocabulary of the most common words in the script. 3,000 tokens have been used in the proposed method. In Word2Vec, a key measure of meaning is the context of the word. Here the built-in word vector model is used to build a semanticized XSS model to allow the system to understand HTML languages like < script > and alerting). For modelling the word2vec class of the genism module is used and the word space is 128 dimensions.

The concept of the neural network in word2vector is to provide our target input words as one-hot vectors. Then we want the neural network to train the probability of a valid context word through a hidden layer while reducing the probability of an invalid context word (i.e. words which never appear in the surroundings of the target word). This includes the use of a softmax on the output layer. When training is completed, the output layer is removed and the weights of the hidden layer are our embedding vectors. A Skip-gram may be depicted as a neural network consisting of a mapping layer and an output layer. Input representation: Every word in the input layer is a single heat encoding, meaning that all words are represented as N-dimensional vectors, where N is the total number of words of the vocabulary. Each word has a size of 1 in the vector and the remainder has a value of 0.

Forward process of propagation in the network: The output layer vector value can be calculated from both the vector of the hidden layer (K dimension) and the weight matrix of the KxN dimension between the hidden layer and the output layer. The output layer is also an N-dimensional vector, which matches a vocabulary word with each dimension. The Softmax activation function is then applied to the vector of the output layer to calculate each word 's probability. Finally, the Softmax is used for calculating the likelihood of any word on the output layer vector.

3.3. CNN based on LSTM

The feature vectores resulted from word2vector step will be using in this step, which consist the proposed method of two deep learning methods which are CNN and LSTM. Furthermore, the CNN based LSTM used to increase the accuracy of our network by using the Convolutionary Neural Network (CNN) one-dimensional three-time and zero padding two-time, resulting from concatenation (CNN1, CNN2 and CNN) use in the LSTM. Figure 3 shows the architecture of CNN-LSTM network phases Figure 3. the architecture of CNN-LSTM.

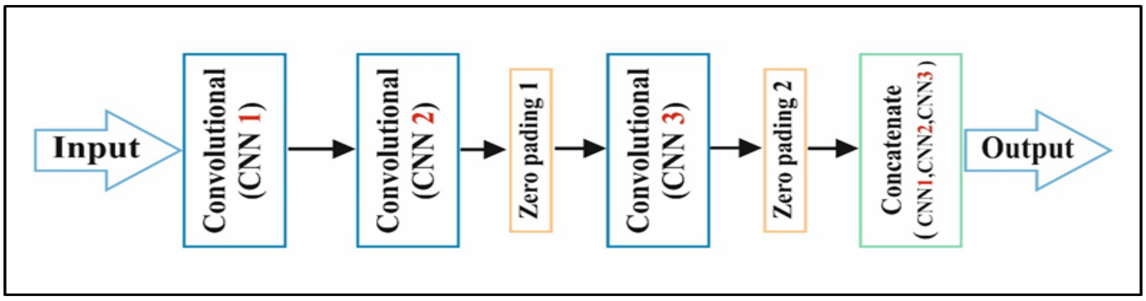

Figure 3. The architecture of CNN-LSTM

The main steps of CNN-LSTM stage above are shown in an algorithm (3):

```
Algorithm 3.3: CNN-LSTM Stage Algorithm
Input     Input Layer
Output    Normal or abnormal
Step1     Convolutional neural network one dimension(CNN1)
Step2     Convolutional neural network one dimension(CNN2)
Step3     Zero padding 1
Step4     Convolutional neural network one dimension(CNN3)
Step5     Zero padding 2
Step6     Concatenate ( CNN1,CNN2,CNN3 )
Step7     LSTM
Step8     Dropout 0.5%
Step9     Fully Connected layer
Step10    Softmax Function
```

4. RESULTS AND DISCUSSION

In this section, the experimental results and evaluation stage have been described.

4.1. dataset setting

The public secuirty dataset is very scarce. The data from the experiment in this proposed method contain two parts which are normal and abnormal, while abnormal samples are more than 40,000 from www.xssed.com and 74,000 normally, to ensure data security the host and path information is removed in the url, and only the payload part is retained. The above data are saved after url encoding in the csv. Since url is encoded for part of the original data, this can only be used after two url decoding.

4.2. Simulation specification

This was conducted on PC with the following features: 2.60 GHz, 16 G RAM and NVIDIA GeForce GTX 1060 with 6 GB, Intel(R) Core(TM) i7-7700 CPU @. Keras is a high-level, Python-developed, neural network API interface that is implementable on CNTK, TensorFlow, and Theano [15]. Being that Kerasis easy to use and exhibits easy extensibility and modularity, I was used in this study with TensorFlow for the development and evaluation of the prposeddetection method.

In this experiment, 10-fold cross validation was used to evaluate the performance of the proposed. First, the dataset was randomly partitioned into 10 equivalent sizes in 10-fold cross validation. Nine subsets were stored as the training set while one set was reservd for the validation of the model. The cross-validation process was executed 10 times, and each of the 10 subsets was used just once as the validation data before averaging to get the final results. The result of each classification can be a binary decision problem be positive or negative. Table 2 showed the representation of the decisionas a confusion matrix in a structure.

Table 2. Confusion matrix

	Actual XSS	Actual Non-XSS
Predicted XSS	TP	FP
Predicted Non-XSS	FN	FP

Moreover, there are four classes in the confusion matrix, these are the True Positives which specifies amount of XSS samples that are correctly classified, the False Positives which portrays the number of Non-XSS samples wrongly classified as XSS, the True Negative that indicate the number of correctly classified Non-XSS samples, and the False Negatives that capture the the number of XSS samples wrongly classified as Non-XSS. We evaluate the experimental results on the basis of a confusion matrix with precise recall, F1 score and the formula criteria in the following equations. The proposed model was objectively evaluaed by comparing the proposed approach to XSS-attacks with the earlier proposed approach by Rathore [5]. Which relies on 10 traditional ML methods to detect XSS.

The performance of the profound learning model is better than ten ML algorithms in Table 3. The proposed model achieved an accuracy rate of 99,3%, recall rate of about 99,1%, F1 score of 99,5%, andprecision value of about 99,9%. Figrepesents the receiver operating characteristic (ROC) curve that captured the detection performance of the proposed approach. The obtained ROC curve was almost perfect, exhibiting a curve area (AUC) of 0.95. Furthermore, the corresponding true positive rate and false positive rate were about 1.0 and 0.01, respectvely. This implies that the proposed deep-learning approach in this study was effective in XSS attacks detection, Figure 4. showed The comparison of the proposed method with the related work.

Table 3. The performance of the profound learning model

Techniques	Recall	Precision	F-measure	Accuracy
RandomForest	0.971	0.977	0.974	0.972
ADTree	0.972	0.970	0.971	0.971
RandomSubSpace	0.969	0.972	0.970	0.970
Decorate	0.968	0.965	0.966	0.966
AdaBoost.M1	0.965	0.964	0.964	0.966
JRip	0.962	0.960	0.961	0.964
NaiveBayes	0.964	0.966	0.965	0.963
SupportVectorMachine	0.958	0.955	0.956	0.958
LogisticRegression	0.955	0.950	0.952	0.956
k-NearestNeighbors	0.950	0.954	0.952	0.947
Proposed approach	0.991	0.999	0.995	0.993

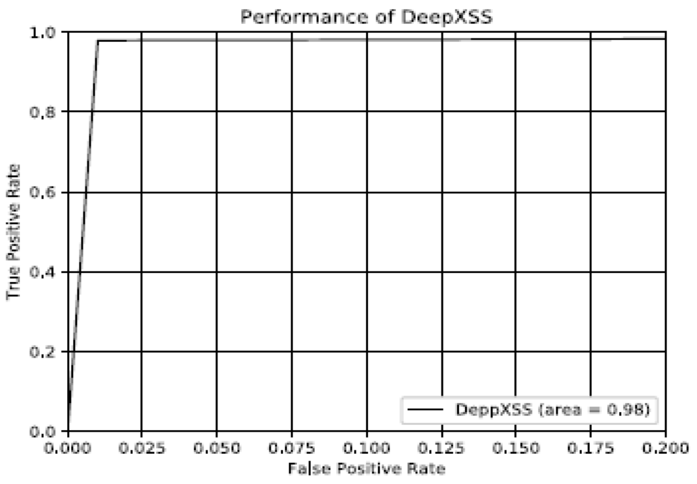

Figure 4. The comparison of the proposed method with the related work

5. CONCLUSION

The detection of cross-site scripts is an essential web security factor. The vulnerability of web applications to XSS attacks can cause theft of user information and other related security issues. In this paper, a deep learning-based approach for XSS attacks detecton was proposed. First, we proposed the restoration of the original data from the obscured version as this is an important preprocessing step;secondly, we utilized word2vec for extraction of the semantic features of the payloads of XSS attacks before mapping them to vectors. Finally, we used CNN with LSTM to build classification models. The suggested solution was tested using 10-fold cross-validation on the real data collection. The evaluation of the prposed approach showed that it achieved an accuracy level of 99,3%, recall rate of 99,1%,precision value of 99,9%, and F1 score of 99,5%.

REFERENCES

[1] Tianlong L., Yu Q., Liang S.andJianan Y., “Locate-Then-Detect: Real-time Web Attack Detection via Attention-based Deep Neural Networks”, *Twenty-Eighth International Joint Conference on Artificial IntelligenceMain track*. Pp. 4725-4731IJCAI, 2018.

[2] Nithya V., Pandian S., Malarvizhi C. “A survey on detectionand prevention of cross-site scripting attack”, *InternationalJournal of Security and Its Applications*, vol. 9, no. 3, pp. 139-152, 2015.

[3] Gupta S., Gupta B.,“Cross-Site Scripting (XSS) attacks anddefense mechanisms: classification and state-of-the-art”, *International Journal of System Assurance Engineering and Management*, vol. 8, no. 1, pp. 512-530, 2017.

[4] Vishnu B A, Jevitha K P.,” Prediction of Cross-Site ScriptingAttack Using Machine Learning Algorithms”, *Proceedingsof the 2014 International Conference on InterdisciplinaryAdvances in Applied Computing*. ACM, p. 55, 2014.

[5] Rathore S, Sharma P K, Park J H.,” XSSClassifier: AnEfficient XSS Attack Detection Approach Based on MachineLearning Classifier on SNSs”, *Journal of InformationProcessing Systems*, vol. 13, no. 4, 2017.

[6] P. Likarish, E. Lung. and I. jo, "Obfuscated malicious javaScript detection using classfication techniques," in *Proceedings of the 4th International Conference on Malicious and Unwanted Software (MAL WARE)*, Montreal, Canada, pp. 47-54, 2009.

[7] M.A. Aluned, and F. Ali, "Multiple-path testing for cross site scripting using genetic algorithms," *Journal of Systems Architecture*, vol. 64, pp. 50-62, 2016.

[8] C. H. Wang andY. S. Zhou, "A new cross-site scriptiogdetecrion mechanism integrated with HTMIL5 andCORS properties by using browser extensions," in *Proceedings of the2016 International Computer Symposium (ICS)*, Chiayi, Taiwan, pp. 264-269, 2016.

[9] H. Shahriar and M. Zulkernine, "S2XS2: A Server SideApproach to Automatically Detect XSS Attacks," *2011 IEEENinth International Conference on Dependable, Autonomicand Secure Computing*, Sydney, NSW, pp. 7-14, 2011.

[10] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, ”A deep learning approach tonetwork intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, p. 4150, 2018.

[11] A. M. Vartouni, S. S. Kashi, and M. Teshnehlab, “An anomaly detectionmethod to detect Web attacks using stacked auto-encoder," in *Proc. 6thIranian Joint Congr. Fuzzy Intell. Syst. (CFIS)*, pp. 131-134, 2018.

[12] K. Grosse, N. Papernot, and P. Manoharan, ``Adversarial examples formalware detection," in *Proc. Eur. Symp. Res. Comput. Secur. Cham, Switzerland: Springer*, pp. 6279, 2017.

[13] S. Rathore, P. K. Sharma, and J. H. Park, “XSSClassifier: An efficient XSS attack detection approach based on machine learning classifier on SNSs,” *J. Inf. Process. Syst.*, vol. 13, no. 4, pp. 1014-1028, 2017.

[14] Stokes, Jack W., Rakshit Agrawal, and Geoff McDonald. "Neural classification of malicious scripts: A study with javascript and vbscript." arXiv preprint arXiv:1805.05603, 2018.

[15] Yanpeng C.,JunjieC., Jianwei H., “A Survey on XSS Attack Detection and Prevention in Web Applications”, *ICMLC: Proceedings of the 12th International Conference on Machine Learning and Computing February*, ACM, pp. 443-449, 2020.

[16] Murugan, Pushparaja. "Implementation of deep convolutional neural network in multi-class categorical image classification." arXiv preprint arXiv:1801.01397, 2018.

[17] Fang, Yong, Yang Li, Liang Liu, and Cheng Huang. "DeepXSS: Cross site scripting detection based on deep learning." In *Proceedings of the 2018 International Conference on Computing and Artificial Intelligence*, pp. 47-51, 2018.

[18] JIANG, Feng, et al. “Deep learning based multi-channel intelligent attack detection for data security”, *IEEE transactions on Sustainable Computing*, 2018.

[19] Castiglione, Arcangelo, Florin Pop, Massimo Ficco, and Francesco Palmieri, eds. “Cyberspace Safety and Security: 10th International Symposium”, *CSS 2018*, Amalfi, Italy, October 29-31, 2018, Proceedings, vol. 11161. Springer, 2018.

[20] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781, 2013.

[21] Z. H. Salih, G. T. Hasan, and M. A. Mohammed, "Investigate and analyze the levels of electromagnetic radiations emitted from underground power cables extended in modern cities," in *2017 9th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, 2017.

[22] Z. H. Salih, G. T. Hasan, M. A. Mohammed, M. A. S. Klib, A. H. Ali, and R. A. Ibrahim, "Study the Effect of Integrating the Solar Energy Source on Stability of Electrical Distribution System," in *2019 22nd International Conference on Control Systems and Computer Science (CSCS)*, pp. 443-447, 2019.

[23] N. D. Zaki, N. Y. Hashim, Y. M. Mohialden, M. A. Mohammed, T. Sutikno, and A. H. Ali, "A real-time big data sentiment analysis for iraqi tweets using spark streaming," *Bulletin of Electrical Engineering and Informatics (BEEI)*, vol. 9, pp. 1411-1419, 2020.

[24] M. A. Mohammed, R. A. Hasan, M. A. Ahmed, N. Tapus, M. A. Shanan, M. K. Khaleel, et al., "A Focal load balancer based algorithm for task assignment in cloud environment," in 2018 10th *International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1-4, 2018.

[25] M. A. Mohammed, A. A. Kamil, R. A. Hasan, and N. Tapus, "An Effective Context Sensitive Offloading System for Mobile Cloud Environments using Support Value-based Classification," *Scalable Computing: Practice and Experience*, vol. 20, pp. 687-698, 2019.