A Comparison of Machine Learning Algorithms for Detecting XSS Attacks

XiaoLong Chen, Mohan Li, Yu Jiang, and Yanbin Sun(✉)

Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China
sunyanbin@gzhu.edu.cn

Abstract. With the rapid increase of web applications, the problem of XSS (cross-site scripting) attacks in Web applications is becoming more and more serious. In the face of more and more complex and changeable XSS attacks, the traditional XSS defense method cannot solve the problem of XSS security, inefficient and accurate recognition effect is poor. Therefore, this paper summarizes the method of XSS recognition based on the machine learning algorithm, classifies different machine learning algorithms according to the recognition strategy, analyzes their advantages and disadvantages, and finally looks forward to the development trend of XSS defense research, hoping to play a reference role for the following researchers.

Keywords: Web applications · XSS recognition · XSS defense · Machine learning

1 Introduction

In the era of explosive growth of Web applications, people get great convenience from Web sites and web apps, but they are also surrounded by web cross-site scripting attacks. In the 2017 Global cybersecurity report released by Trustwave, the most common types of cyberattacks in 2016 were disclosed, with XSS attacks accounting for 20.1% of the top ten total attacks, accounting for 13% of all attacks, and in all high-risk vulnerability statistics, the number of XSS vulnerabilities ranked first with 29.6%, which is enough to show that the current XSS attack situation is very serious. And once a user or website is attacked by XSS, the harm is considerable. For example, hackers can steal users' cookie through XSS malicious scripts and steal user accounts, make illegal transfers, etc. Or they can control enterprise data through XSS malicious scripts, such as reading, deleting, tampering, adding enterprise-sensitive data, and even using XSS malicious scripts combined with other vulnerabilities to implement DDoS attacks on websites. The consequences of these hazards are serious, so researchers have taken a series of methods to defend against XSS attacks.

X. Sun et al. (Eds.): ICAIS 2019, LNCS 11635, pp. 214–224, 2019.
https://doi.org/10.1007/978-3-030-24268-8_20

2 Traditional XSS Defense

Traditional XSS defense is mainly based on two ideas, one is to filter the user's input, the other is to escape the content that is output to the website. Input filtering for users is mainly based on pattern matching. Pattern matching is the most commonly used way to detect web malicious code, according to the filtering idea can be divided into blacklist filtering and whitelist filtering. As the name suggests, blacklist filtering is the filtering of input that can be a threat to websites and users, such as XSS malicious code that typically implements its malicious functionality through JavaScript scripts, so we can match the keywords such as <script>, alert, prompt, document, cookie and other characters in the input, then delete or replace the keywords, so malicious XSS code cannot launch attacks.

The whitelist filtering method is the same, but the different idea is to keep only the keywords and characters required by the site, for example, the input requirements of the site can only be 0 to 9 digits and A to Z lowercase letters, the rest of the characters cannot be entered, then even the normal input containing other characters will be filtered out. Of course, this filtering method can largely avoid XSS attacks, but it also affects the function and scope of application of the website to a certain extent.

The idea of output escaping and input filtering is similar. The difference is that output escaping is character escaping the content of the user input that will be displayed on the page or executed on the browser, and the sensitive characters that may trigger XSS are transferred or encoded into normal characters, so that XSS malicious code cannot be executed. And keyword filtering is to perform feature matching before user input enters the website, so output escaping can be used as a supplementary measure for keyword filtering, for unfiltered XSS attack statement escaping can avoid XSS attacks to a greater extent.

3 Defects of Traditional XSS Defense

Although the traditional defense measures based on filtering rules and output escape can avoid XSS attacks to some extent, the disadvantages are also obvious: The XSS attack statement is flexible, the speed of filtering rule library update is difficult to keep up with the change speed of XSS attack statement combination rules, It is extremely time-consuming and error-prone to manually discover the keyword combination rules for new XSS attack statements. And when the output is escaped, the way to escape is determined based on the context information of the current location, and it is a tedious task to decide how each location should be escaped in the face of so many and complex pages. And it is not the site security maintenance personnel who decide how to escape, but the specific business personnel (because the business personnel decides how the current location should be displayed), so it is difficult for all business personnel to understand the XSS statement rules. If not handled properly, it will generate more XSS vulnerabilities, so it is not an easy task to determine the escaping rules completely manually.

Just as Qiu, et al. [1] faced heterogeneous data in city big data, XSS data is also becoming more and more heterogeneous, for example, XSS statements are hidden in

URL links, pictures and even script files, which are difficult to be recognized by traditional methods. Traditional XSS defenses are also less effective against XSS attacks combined with other means, such as ransomware and RDF protocols [2].

In order to automatically identify XSS attacks and perform XSS defenses, web security researchers have proposed XSS recognition technology based on machine learning.

4 Overview and Suggestions on the Application of Machine Learning in XSS Detection

Machine learning algorithms applied to XSS recognition have been researched since long ago, but early research is more focused on the detection of malicious web pages. For example, Cohen [3] applied the decision tree algorithm to the detection of Web pages in 1996. Kan and Thi [4] was also one of the first researchers to apply machine learning algorithms to malicious Web page detection, and their work in 2005 focused on keywords in URLs and their location in URLs. Ma, Saul, Savage, and Voelker [5] focus on using online learning to detect malicious Web pages from URLs features. In the research work of Kazemian, Ahmed [6], for the first time, they applied unsupervised machine learning algorithms k-means and affinity propagation to the detection of malicious Web pages. In the study of Krishnaveni and Sathiyakumari [7], they used naive Bayesian [8], decision trees, and multi-layer perceptron (MLP) to classify attacks containing XSS Web pages. Wu and Lin [9] use hidden markov models to detect cross-site scripting attacks, and compared with logistic regression and naive bayesian algorithm, achieved better accuracy and recall rate. Vishnu and Jevitha [10] used support vector machine, J48 decision tree and bayesian algorithm to predict the cross-site scripting attack. Zhang [11] explored a new method for finding frequent itemsets of eigenvectors of XSS attacks using apriori and FP-growth algorithms. Liu, Fang and Liu [12] and others use deep learning to detect cross-site scripting attacks, and use the Word2vec word vector model to deal with XSS features, compared with the traditional machine learning algorithm ADtree and AdaBoost algorithms used by Wang [13], better accuracy and recall rates have been achieved. Next, this paper will briefly analyze the processing flow of each machine learning algorithm in XSS detection and its advantages and disadvantages.

In addition, the work done by the above researchers is more reflected in the passive defense of XSS. For active defense of XSS, just like the method proposed by Chen, et al. [14], which automatically mines security-sensitive functions from source code, we can also consider automatically mining security-sensitive functions of Web programs and correcting them before attackers attack. For the XSS adversarial examples attacks on the trained model, as Zeng, et al. [15] used adversarial learning for distant supervised relation extraction, we can also conduct adversarial learning on the machine learning model based on the generative adversarial networks before the attacker attacks, so as to improve the robustness of XSS detection model.

5 Application Process of Different Machine Learning Algorithms in XSS Detection

This section will briefly analyze how each machine learning algorithm is applied to the detection and recognition of cross-site scripting attacks, and the advantages and disadvantages of each machine learning algorithm in cross-site scripting attack recognition processing.

5.1 Naive Bayes

The naive bayesian algorithm is based on Bayesian theorem, and the naive bayesian classification is the simplest and common classification method in Bayesian classification. Bayesian theorem is as follows:

$$\mathrm{P(B|A)} = \frac{P(A|B)P(B)}{P(A)} \tag{1}$$

As a Bayesian rule based on statistical method, the naive bayesian hypothesis has an important assumption that each feature is independent of each other, and it classifies by calculating the probability and cost associated with each decision. Specifically, for a data to be classified, the posterior probability of each class is calculated, and the data to be classified belongs to the class with the highest posterior probability. In the training process, each observation sample in the training set can incrementally increase or decrease the probability of hypothesis occurrence [16]. The naive bayesian algorithm can usually achieve high classification accuracy and its calculation cost is relatively small, but as mentioned earlier, the naive bayesian algorithm is based on the premise that each feature is independent of each other, and the characteristics of cross-site scripting attack statement are usually closely related. Therefore, this will affect the recognition accuracy of Naive Bayes algorithm for cross-site scripting attacks to a certain extent, and in the research of Nunan, et al. [17], the overall performance of SVM in cross-site scripting attack recognition is better than that of the Naive Bayesian algorithm.

Moreover, when the naive bayesian classifier trains according to XSS samples, it may have unbalanced XSS data, which is also the problem faced by all XSS classifiers in training. Some effective sampling algorithms, such as bidirectional self-adaptive resampling algorithm [18], can be adopted to mitigate the impact on the model detection effect to a certain extent.

5.2 K-means

K-means Algorithm is an unsupervised clustering algorithm, which is a typical representative of the target function clustering method based on prototype. Prototype-based clustering algorithm assumes that the clustering structure can be characterized by a set of prototypes, which is very common in real clustering tasks. K-means need to determine a parameter K, which indicates the number of clusters generated by K-means, and in calculating the similarity between the cluster and the cluster, K-means

takes the European distance as the similarity measure, which is to find the optimal classification corresponding to an initial clustering center vector V, so that the evaluation index J is the smallest. The algorithm uses the sum of squares of errors as the clustering criterion function. Specifically, given the sample set $D = \{x_1, x_2, \ldots x_m\}$, k-means algorithm divides clusters of $C = \{c_1, c_2, \ldots c_k\}$ to minimize squared error [19]:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2 \tag{2}$$

In order to minimize the square error, the K-means algorithm uses greedy strategy to approximate the squared error by iterative optimization.

Shar, Tan [20] proposed a cross-site script attack prediction model based on classification and clustering. They used the research content of cluster-based intrusion detection of Portnoy [21], that is, the optimal parameter N (the percentage of maximum clustering) containing a large number of vulnerable clusters is 15%. In such a K-means clustering model, their predictors average 76% recall rate and 39% accuracy, indicating that they are effective without labeled training data. Although it is not as accurate as supervised learning, it is much easier to construct an unsupervised model such as k-means clustering than to construct a supervised model for attack detection.

5.3 Decision Tree

Decision tree is a kind of supervised learning algorithm, which is widely used as a classification method. According to the structure of the decision tree, the decision tree can be divided into a binary decision tree and a multi-fork tree. For example, some decision tree algorithms only generate a binary tree (where each internal node just branches out two branches), while other decision trees may generate a non-binary tree, where each node represents an attribute, the branch path is selected according to the value of the attribute, and the leaf node represents the result of the classification.

The training of decision tree algorithms for identifying cross-site scripting attack statements consists of two main processes: construction and pruning. First of all, according to the various feature attributes in the attack statement to construct the decision tree, such as branch determination based on the value of some sensitive keywords (including Script, document, alert, etc.); then pruning (specifically divided into pre-pruning and post-pruning) gives "subtraction" to branches that have little effect on the classification results, which can reduce the amount of calculation or increase the accuracy of classification.

Krishnaveni and Sathiyakumari [7] use decision tree algorithm to identify cross-site scripting attacks, and achieve the same good recognition effect as multi-layer perceptron (MLP). In Vishnu and Jevitha's [10] experiments, the decision tree algorithm has the same performance as Bayesian and SVM in correctly identifying cross-site scripting attacks, and it is less easy to identify attack statements as benign ones, and the time taken for model generation is less than SVM but more than Bayesian algorithm.

5.4 Association Rules

Association rule analysis is an algorithm that discovers interesting associations and related relationships between itemsets from large amounts of data, and a classic application of association rules is the placement of supermarket shelves (such as the famous story of beer and diaper). There are two important concepts in the association rule algorithm, one is the support, the other is the confidence, the support indicates that the dataset contains the proportion of an item set record, the specific formula is as follows:

$$\text{Support}(\text{X} => \text{Y}) = \frac{\textit{Number of records containing both X and Y}}{\textit{Total number of data sets recorded}} \tag{3}$$

Confidence indicates the probability that Y is pushed out by x in the case where the precondition x occurs. A simple understanding is a conditional probability. The specific formula is as follows:

$$\text{Confidence}(\text{X} => \text{Y}) = \frac{\textit{Number of records containing both X and Y}}{\textit{The data set contains the number of records of X}} \tag{4}$$

The common association rule algorithm are Apriori algorithm and FP-Growth algorithm. Apriori algorithm is a traditional association rule algorithm, and when using Apriori algorithm to mine the association rules of cross-site script attack statement, it is usually necessary to split the statement according to some special characters, and then the association of these separated parts is calculated, the following steps are mainly used when calculating the association:

1. According to the minimum support, the words satisfying the support degree are selected from the segmented word set to form a frequent set;
2. Combine the results from the first step to form a candidate set, and add only one word that is not in the original item set for each combination;
3. Scan the word set again, select the two-two combination that satisfies the support, and get a new frequent set;
4. Frequent sets are combined again. Each combination only adds a word that is not found in the original item set to form a new candidate set;
5. Scan the initial set of words, select the item sets that satisfy the support from the candidate set, and form a new frequent set;
6. Repeat the above steps to finally obtain a frequent set with strong correlation, select the item sets with strong associations according to the confidence in frequent set, and obtain the strong correlation between the words in the statement.

And the part with strong correlation constitutes the feature rule of the statement together. But the Apriori algorithm scans the whole dataset every round, and the efficiency is very low, so the FP-Growth improves the algorithm and only scans two times in the process of operation, which greatly improves the efficiency. The strong association relationship of each part of the statement obtained by association rule can be used as an important content of feature extraction in other algorithm operation.

5.5 Support Vector Machine

Support Vector Machine (SVM) is a two-class model, which divides the dataset by looking for a hyperplane, and in order to make the classification result better, the principle is to maximize the separation of the data set. Specifically in the classification of cross-site scripting attack statements, their intuitive representation is as follows (Fig. 1):

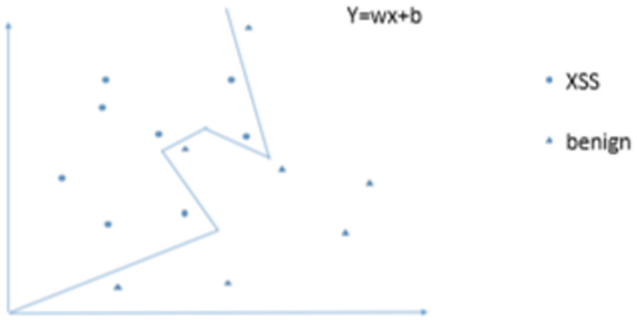

Fig. 1. SVM model diagram

For Web access datasets, if the normal statement and the cross-site scripting attack statement can be clearly distinguished after feature vectorization, the training process of the SVM is very simple, and the attack statement can be separated from the normal statement by a simple plane. But in reality normal statements and attack statements are usually nested with each other, there is no obvious distinction, so the normal statement and attack statement cannot be separated by a simple linear plane, and it is necessary to map the statement data which cannot be separated by linear plane into the high dimensional space through the very important kernel function in SVM, and then divide it through the hyperplane. The choice of kernel function will directly affect our final classification results. In the experiment of Choi, Choi, Ko [22], the data set and code dictionary obtained by using n-Gram generated malicious code and pattern matching are applied to the SVM classifier, and the cross-site scripting attack code was effectively detected, which achieved better performance than the naive bayesian and keyword mode methods.

5.6 Hidden Markov Models

The Hidden Markov model (HMM) is a statistical model used to describe a Markov process with implicit unknown parameters, the difficulty of which is to determine the implicit parameters of the process from observable parameters, and then use these parameters for further analysis, such as pattern recognition. Further understanding is actually the simplest dynamic Bayesian network, which is used for modeling time series data. The graph model structure is as follows (Fig. 2):

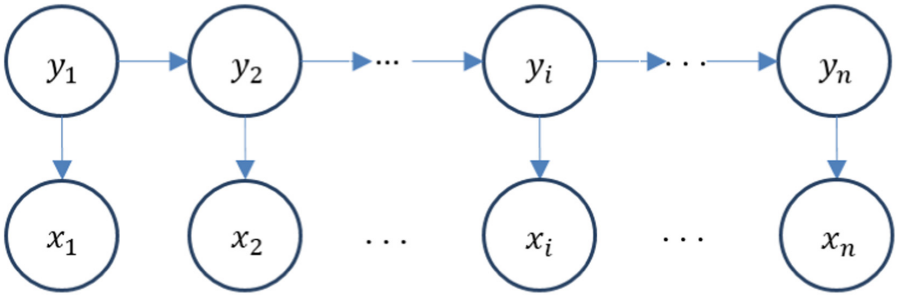

Fig. 2. Markoff model diagram

The value of the observed variable of the hidden Markov model depends only on the state variable, that is, the x_t is determined by y_t, and the state y_t at time t only depends on the state y_{t-1} at time t − 1, that is, the state of the next moment of the system is determined only by the current state and does not depend on any previous state. Based on this dependency, the combined probability distribution of all variables is [19]:

$$P(x_1, y_1, \ldots, x_n, y_n) = P(x_1|y_1) \prod_{i=2}^{n} P(y_i|y_{i-1}) P(x_i|y_i) \quad (5)$$

When Wu Jr [9] use the above hidden Markov model to detect cross-site script attack statements, the HTTP request of the statement is first converted into a token sequence, and then the time relationship in the token sequence is modeled by HMM model to determine the XSS attack. Specifically, the token sequence extractor converts HTTP requests into token sequences. The HMM-based token correlator uses HMM to extract the token correlation of adjacent tokens in the token sequence, and the XSS attack detector is responsible for determining whether the input token sequence contains any XSS attacks and where the attacks are. The experimental results show that 100% of all cross-site script attacks can be identified, and the rate of misidentification (recognizing benign statements as attack statements) is only 0.3%, which has a very good recognition effect.

5.7 Deep Learning

The concept of deep learning originates from the research of artificial neural network, and the multi-layer perceptron (MLP) with multi-hidden layer is a kind of deep learning structure, and the typical deep learning model is a deep neural network. Hinton [23] in the Deep Belief network (DBN) proposed the non-supervised greedy layer-by-layer training algorithm, which helps to solve the optimization problems related to deep structure, and this is the core algorithm of deep learning training. Specifically, every time a layer of hidden nodes is trained, the output of the previous layer of hidden nodes is used as input, while the output of the current layer of hidden nodes is used as input of the next layer of hidden nodes. Finally, the whole network is trained with BP algorithm [24] (Fig. 3).

Using deep learning to detect cross-site scripting attacks, it is necessary to vectorize the sample statements and then train the deep learning algorithm model. When using the trained model to predict the newly entered statements, it is also necessary to carry out the same vectorization, through the calculation of the model, to determine whether it is an attack statement. The figure is as follows:

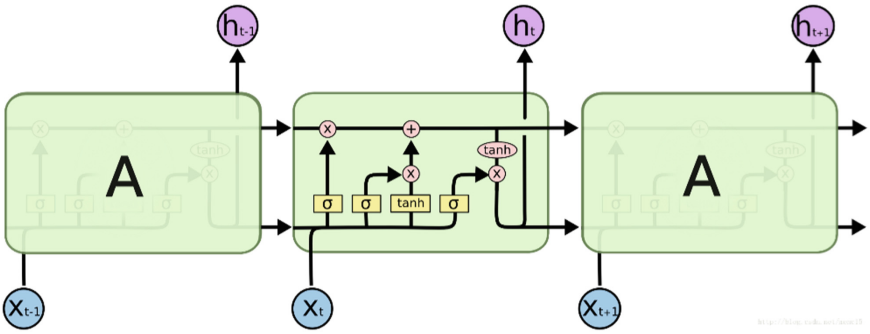

Fig. 3. LSTM model diagram

In the experiment of Liu, Fang and Liu [12], they used word2vec [25] to generate word vectors, Word2vec was a software tool developed by Google to train word vectors, and the word vectors generated by Word2vec were able to preserve the semantic information of words. Therefore, the semantic relations of each part of attack statements are also preserved, and the traditional machine learning algorithm cannot achieve this. And finally use the long-length memory (LSTM) recurrent neural network to establish the classification model, the final accuracy of the model is 99.5%, and the recall rate is 97.9%, which is a very good detection effect.

6 Conclusion

This paper mainly introduces the application of different machine learning algorithms in the detection and recognition of cross-site scripting attacks, including the classical unsupervised machine learning algorithms, such as Association rule algorithm(Apriori) and K-means clustering algorithm, as well as support vector machine, decision tree, naive bayesian and other supervised learning algorithms, and the most popular deep learning algorithms at the moment, such as lstm (long time memory model). Each machine learning algorithm has different application scenarios when detecting attacks, in different cases to choose the appropriate algorithm can achieve the best results. I hope this paper can bring some help to the following researchers.

Because this paper mainly introduces the application of each algorithm in cross-site scripting attacks detection and recognition, and does not introduce the situation of their combination use, such as the combination of Apriori algorithm and other algorithms, so the next important research direction is the machine learning algorithm combined to

detect cross-site scripting attacks. Moreover, the machine learning algorithm introduced in this paper is mainly used in the recognition of cross-site scripting attacks, and does not explore how to extract the characteristics of attack statements more intelligently and efficiently, so it is also an important research direction to study how to extract the feature rule of XSS attack statements accurately and efficiently by machine learning algorithm.

Acknowledgments. This work is funded by the National Key Research and Development Plan (Grant No. 2018YFB0803504) and the National Natural Science Foundation of China (No. U1636215).

References

1. Qiu, J., Chai, Y., Liu, Y., et al.: Automatic non-taxonomic relation extraction from big data in smart city. IEEE Access **6**, 74854–74864 (2018)
2. Wang, Z., Liu, C., Qiu, J., et al.: Automatically traceback RDP-based targeted ransomware attacks. Wirel. Commun. Mobile Comput. (2018)
3. Cohen, W.W.: Learning trees and rules with set-valued features. In: AAAI/IAAI, vol. 1, pp. 709–716 (1996)
4. Kan, M., Thi, H.: Fast webpage classification using URL features. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, pp. 325–326. ACM (2005)
5. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Identifying suspicious URLs: an application of large-scale online learning. In: Proceedings of the 26th Annual International Conference on Machine Learning, ICML, pp. 681–688 (2009)
6. Kazemian, H.B., Ahmed, S.: Comparisons of machine learning techniques for detecting malicious webpages. Expert Syst. Appl. **42**(3), 1166–1177 (2015)
7. Krishnaveni, S., Sathiyakumari, K.: Multiclass classification of XSS web page attack using machine learning techniques. Int. J. Comput. Appl. **74**(12), 36–40 (2013)
8. Bayes, T., Bayes, T.: An essay towards solving a problem in the doctrine of chances. Resonance **8**(4), 80–88 (2003)
9. Wu Jr, Y.T., Lin Jr, S.J., Liu Jr, E.S., et al.: Cross-site scripting attack detection based on hidden Markov model (2009)
10. Vishnu, B.A., Jevitha, K.P.: Prediction of cross-site scripting attack using machine learning algorithms. In: Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing. ACM (2014)
11. Zhang, W.: Research on XSS vulnerability detection model based on feature injection. Lanzhou University of Technology (2016)
12. Fang, Y., Li, Y., Liu, L., et al.: DeepXSS: cross site scripting detection based on deep learning. In: Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, pp. 47–51. ACM (2018)
13. Wang, R., Jia, X., Li, Q., et al.: Machine learning based cross-site scripting detection in online social network. In: 2014 IEEE International Conference on High Performance Computing and Communications, 2014 IEEE 6th International Symposium on Cyberspace Safety and Security, 2014 IEEE 11th International Conference on Embedded Software and System (HPCC, CSS, ICESS), pp. 823–828. IEEE (2014)
14. Chen, L., Yang, C., Liu, F., et al.: Automatic mining of security-sensitive functions from source code. Comput. Mater. Continua **56**(2), 199–210 (2018)

15. Zeng, D., Dai, Y., Li, F., et al.: Adversarial learning for distant supervised relation extraction. Comput. Mater. Continua **55**(1), 121–136 (2018)
16. Alpaydın, E.: Introduction to Machine Learning, 2nd edn. The MIT Press, Cambridge (2010)
17. Nunan, A.E., Souto, E., Dos Santos, E.M., et al.: Automatic classification of cross-site scripting in web pages using document-based and URL-based features. In: 2012 IEEE Symposium on Computers and Communications (ISCC), pp. 000702–000707. IEEE (2012)
18. Han, W., Tian, Z., Huang, Z., et al.: Bidirectional self-adaptive resampling in internet of things big data learning. Multimedia Tools Appl. 1–16 (2018)
19. Zhou, Z.: Machine Learning, 1st edn. Tsinghua University Press, Beijing (2016)
20. Shar, L.K., Tan, H.B.K., Briand, L.C.: Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis. In: Proceedings of the 2013 International Conference on Software Engineering, pp. 642–651. IEEE Press (2013)
21. Portnoy, L., Eskin, E., Stolfo, S.: Intrusion detection with unlabeled data using clustering. In: ACM CSS Workshop on Data Mining Applied to Security (2001)
22. Choi, J.H., Choi, C., Ko, B.K., et al.: Detection of cross site scripting attack in wireless networks using n-Gram and SVM. Mobile Inf. Syst. **8**(3), 275–286 (2012)
23. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
24. McClelland, J.L., Rumelhart, D.E., PDP Research Group.: Parallel distributed processing. Explor. Microstruct. Cogn. **2**, 216–271 (1986)
25. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)