RESEARCH ARTICLE OPEN ACCESS

SOFTWARE VULNERABILITY CLASSIFICATION MODEL USING NEURAL NETWORK

Ms. N. Zahira Jahan M.C.A., M.Phil.,[1], S. Madeshwaran[2]

[1]Associate Processor, Department in Computer Applications, Nandha Engineering College (Autonomous), TamilNadu, India

[2]Final MCA, Department of Computer Applications, Nandha Engineering College (Autonomous), Erode, Tamilnadu, India.

Email: [1]zahirajahan1977@gmail.com, [2]madhumadhu9323@gmail.com

Abstract. Security risks are caused mainly due to software vulnerabilities. If any vulnerability is exploited due to a malicious attack, it will greatly compromise the system's safety. It may even create catastrophic losses. So, automatic classification methods are enviable for effective management of vulnerability in software, thereby improving security performance of the system. It will reduce the risk of system being attacked and spoiled. In this study, a new model has been proposed named automatic vulnerability classification model (IGTF-DNN) Information Gain based on Term Frequency - Deep Neural Network. The model is built using information gain (IG) which is based on frequency-inverse document frequency (TF-IDF), and deep neural network (DNN): TF-IDF is used for calculating frequency/weight of words prepared from vulnerability description; Information Gain is used to select features for gathering optimal set of feature words. Then deep neural network model is used to construct an automatic vulnerability classifier to achieve effective vulnerability classification. The National Vulnerability Database of the United States has been used to test proposed model's effectiveness. Compared to KNN, the TFI-DNN model has achieved better performance in evaluation indexes which includes precision and recall measures.

Keywords: Software Engineering, Software Vulnerability, Deep Neural Network, Information Gain.

I. INTRODUCTION

Due to information technology's fast development, its impacts to industries by application of the Internet and computers are powerful. Not only, they brought convenience, but also huge risks and hidden dangers at the same time. The digitalization level of industries' improvement makes information security problems to become increasingly outstanding. Vulnerabilities are software/hardware defects and problems of system being illegitimately exploitable made by people who are unauthorized nature. As soon as vulnerability of information system is exploited by suspicious attack, the information system's security will be at great risk. It may even create inestimable consequences. In 2017, Windows system vulnerabilities are exploited by hackers to expose 100,000 organizations around the world to Bitcoin ransomware. Again in the same year, Microsoft released a total of 372 vulnerability patches for Office.

Hackers make use of office vulnerabilities to conduct Advanced Persistent Threat (APT) attacks, spread ransomware, botnets and so on. Nowadays, the count and variety of vulnerabilities are gradually increasing, so that the analysis and management of software vulnerabilities are becoming more important.

If the vulnerability can be classified and managed with effectiveness, it may not only enhance the efficiency of vulnerability recovery and management, but also diminish the risk of systems being attacked and collapsed, which is crucially important for security performance of systems. As software security vulnerabilities play a major role in cyber-security assault, more and more researches on vulnerability classification are conducted by applicable security researchers.

The earlier vulnerability classification method RISOS [1], is aimed at the operating system of computer, mainly segments the OS vulnerabilities into 7 categories from the attack perspective, and elaborates how to exploit vulnerabilities instead of triggering the vulnerabilities' situation. The PA vulnerability classification method in [2] not only studies the operating system vulnerabilities, but also classifies the vulnerabilities already present in the application.

Andy Gray vulnerability classification method [3] introduced a vulnerability classification system consisting often categories according to the various analysis needs of vulnerability. As the complexity of vulnerabilities increases, limitations of traditional artificial vulnerability classification methods become clearer.

Therefore, researchers give more attention to vulnerabilities' automatic classification. Recently, a large number of machine learning methods have been reported in text classification field [4]. Classifying data by vulnerability description is a kind of text classification.

Therefore, the automatic classification of vulnerabilities problems can be solved using machine learning methods. Shua at al. [5] applied the SVM classification method based on LDA model in vulnerability classification domain.

The SVM based upon topic model makes full use of number of distributed vulnerabilities for classification. The experiment results indicated that SVM has attained good results in vulnerability grouping.

Wijayasekara et al. [6] evaluated Naïve Bayes classification method by using textual information from error descriptions. That analysis illustrated the feasibility of Naïve Bayes classifier for classifying textual information based on vulnerability description.

Sarang et al. [7] introduced a classification method to classify CVE entries which could not give enough information into vulnerability groups using Naïve Bayes classifier. Gawronetal [8] pertained Naïve Bayes algorithm and simplified artificial neural network (ANN) algorithm for vulnerability classification, and thereby made comparison on same data set. The experimental results verified that artificial neural network algorithm was finer to Naive Bayes algorithm in vulnerability classification.

II. LITERATURE REVIEW

Even though these machine learning algorithms have achieved hopeful results in many fields, because of the huge amount of vulnerability data with short description, generated word vector space handed the characteristics of high dimension and sparse. These machine learning algorithms are not much efficient dealt with high / sparse problems. Meanwhile, they paid no attention to particular vulnerability information and so classification accuracy was not elevated. In recent years, deep learning found application in variety of fields and has achieved triumph, such as the speech and image recognition field [9], [10], and there, the error rate in speech recognition is lowered by 20 – 30 percent [11]. The error rate in ImageNet evaluation task is lowered by 26 – 15 percent [12]. Deep learning also has a important impact in the natural language filed [13], [14].

Jo et al. [15] studied the classification problems in the natural language field, and they applied convolutional neural networks (CNN) and recurrent neural networks (RNN) for large-scale text classification and achieved success.

Aziguli et al. [16] introduced a new text classifier using DNN model to progress the computational performance of processing large text data along with mixed outliers. Therefore, to better deal with the high and sparse word vector space and thereby take benefits of automatic feature extraction by deep learning, this paper introduces an automatic vulnerability classification model IGTF-DNN based on term frequency-reverse document frequency (TF-IDF), information gain (IG) and deep neural network (DNN).

In the model, IGTF algorithm is first used to grab the feature of description text and reduce the generated high-dimensional word vector space dimension. Then a DNN neural network model (based on deep learning) is constructed. The model was trained and tested with vulnerability data taken from National Vulnerability Database (NVD). The test results showed that the automatic vulnerability classification model in this paper effectively improves the performance of vulnerability classification.

The remaining of this paper is organized as follows. This section discusses below the definition of relevant algorithms. Section 3, described the implementation details of the model. Section 4 discussed the experiment dataset and results with comparative analysis. Section 5 outlines experimental procedure and Section 6 outlines the conclusions and possible future research.

The automatic classification model of vulnerability (IGTF-DNN) based on TF-IDF is constructed in this paper. The relevant definitions are as follows.

A. TF-IDF (Term Frequency/Inverse Document Frequency) is a common weighted technology which is found out based on statistical methods [17]. For example, consider there are a set of documents and each document contains a number of terms/words. It is defined that the word I's importance in document j as follows.

$$Tf_{ij} = n_{i,j} / \sum_k n_{k,j} \quad (1)$$

where both I and j are positive integers, ni,j denotes the term I's frequency in document j.

The IDF formula is as follows.

$$idfi = \log (|F| / |\{ j : t_i \in d_j \}|) \quad (2)$$

where |F| is the total number of documents in corpus, f_j is the j_{th} document, and $|\{j:t_i \in f_j|$ is the number of documents containing the term ti.

The TF-IDF formula is as follows.

$$TF{-}IDF = tf_{ij} * idf_i \quad (3)$$

TF-IDF is used to measure the terms' importance word to a document in the document set o rin a corpus. The terms' importance increases proportionally with number of times it appears in the document, but also decreases inversely with frequency it appears in corpus.

B. INFORMATION GAIN (IG) refers to that, if a feature X in class Y is known already, information uncertainty of class Y decrease, and so reduced uncertainty degree will reflect importance of feature X to class Y. Set the training data set to D, |D| shows the count of samples in D. Suppose there are K classes Ck, k = 1, 2, … , K $|C_K|$ is the count of samples fit in to class Ck. $\sum_{Ck=1}^{K} |C_K| = |D|$. If feature A has n different values $\{a_1, a_2, \ldots, a_n\}$, D is segmented into 'n' sub groups according to feature A values, represented as D = (D1, D2, … , Dn), where $|D_i|$ is the samples count in Di, $\sum_{i=1}^{n} |Di| = |D|$. The samples set fit into class C_k in D_i is D_{ik}, $D_{ik} = D_i \cap D_k$, $|D_{ik}|$ is the samples count of D_{ik}.

The empirical entropy H (D) of data set D is calculated as follows.

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \quad (4)$$

The empirical conditional entropy H(D|A) of featureA for dataset D is calculated as follows

$$H(D|A) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|} \quad (5)$$

The information gain calculation formula for each feature is as follows.

$$g(D, A) = H(D) - H(D|A) \quad (6)$$

Based on feature selection method of information gain criterion, each feature's information gain is measured, and the features with larger IG value are selected.

III. THE IGTF-DNN ALGORITHM

The vulnerability automatic classification model IGTF-DNN is composed of IG, TF and DNN. The original vulnerability data is first preprocessed, and then TFI is used to grab features of vulnerability description text and lower the dimensionality of generated higher-dimensional word vector space, and later the DNN is build to comprehend automatic training and classification of vulnerability.

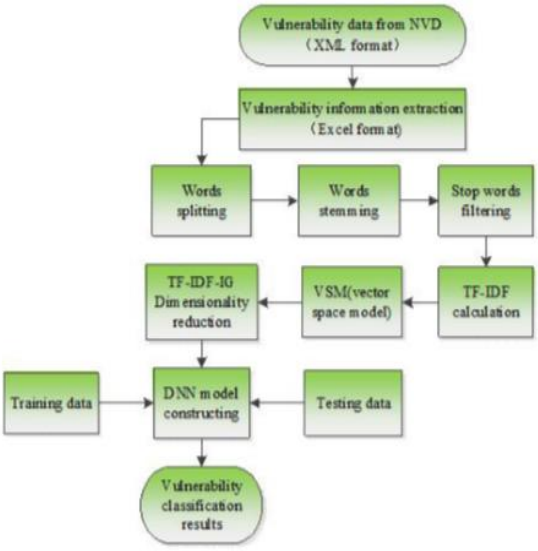

Figure 1. TFI-DNN algorithm

A. FEATURE SELECTION USING TFI: TFI is used to grab feature word set. The feature selection steps are shown in Algorithm 1.

B. OPTIMIZATIONS USING DNN: DNN includes single input layer, one hidden layer (can be made to two if required) and single output layer, whose input is the instance's feature vector and output is instances' category. It includes one propagation process, forward propagation and back propagation. The process of propagation is shown in Algorithm 2.

IV. EXPERIMENTAL ENVIRONMENT AND DATA SET

A. ENVIRONMENT:

The experiment was conducted on PC with Intel(R) Core(TM) i3 processor, 2.4 GHz and 8.00 GB memory, running Windows 10 operating system. Programming uses R 3.4 on R Studio version 0.99.

B. DATA SET OF EXPERIMENT:

In order to test effectiveness, internationally recognized National Vulnerability Database (NVD) [19] is used as experimental data. The source file of this dataset is a series of records taken into excel worksheet, which contains information about vulnerability, such as 'CVE number', 'vulnerability release date', 'CVSS_version', 'type' and 'vulnerability text description' of Vulnerability type and description are taken for analysis.. The annual vulnerability amount (2015-2019) NVD vulnerability database is taken and the record sample count is 600.

The required vulnerability information is extracted from the records using program codes written in R including vulnerability text description, and vulnerability type (category).

Some text information of vulnerabilities from 2015 to 2019 is collected for statistics, 500 of them were used for training data set and 100 for test data set.

Algorithm 1

Input:

Word list(word_list) formed by term document matrix and stop word filtering.

Output:

Feature word set (feature_words).

1) Traversing each word in the word_list.

2)Word frequency statistics for word_list, stored in the doc_frequency list.

3) Traversing the word frequency list doc_frequency.

4) Calculate the TF value of each word according to (1) and store it in the word_tf dictionary.

5) Calculate the IDF value of each word according to (2) and store it in the word_idf dictionary.

6) Calculate the TF-IDF value of each word according to (3) and store it in the word_tf_idf dictionary.

7) The word set is sorted in descending according to the TF-IDF value.

8) Select the first n words as an important feature set.

9) Save important words in the feature list (features_ vocabSet).

10) Traverse features_vocabSet, divide features_vocabSet and store the subset into the subDataSet.

11) Calculate probability of subDataSet.

12) Calculate the empirical conditional entropy of each word according to (4) and (5) and store it in newEntropy.

13) Calculate the IG value of each word according to (6).

14) Save each word and the corresponding IG value in the dictionary.

V. EXPERIMENTAL PROCEDURE

A. DATA PREPROCESSING

Excel work sheet records are taken from 5 excel work sheets for year 2015, 2016, 2017, 2018 and 2019. They are stored in data frame objects. Then category array is filled with unique values of categories from

Vulnerability type column. Separate data frames are formed from subsets which store each category. Term Document Matrix is found and all the terms are stored in array. Stop word removal and punctuation removal is carried out using tm package's tm_map method. Out of 2237 words/terms extracted 113 terms are found as unique and they are taken for further analysis. These numbers may vary if more vulnerability description samples are taken as dataset.

B. MATRIX FORMATION

TF Matrix, IDF Matrix and TF-IDF array is found out using the equation given above (1 to 3) and the terms are sorted based on TF-IDF values.

C. INFORMATION GAIN

Then entropy H (D) of data set D is calculated and later Information Gain values are calculated for each feature and then sorted descending based on IG values.

Sample TF-IDF features:
tfidf_df_sorted:

	Word	Freq
1	xss	0.005312172
2	service	0.005032584
3	via	0.004962687
4	denial	0.004892790
5	web	0.004683099
6	attackers	0.004543305
7	remote	0.004473408
8	cause	0.004473408
9	scripting	0.004403511
10	crosssite	0.004333614

OutputDF:
Feature with GainValue

	Feature	GainValue
1	xss	0.018107652
2	service	0.106077030
3	Via	0.735451987
4	Denial	0.081597715
5	Web	0.090538261
6	Attackers	0.610320229
7	remote	0.606898196
8	cause	0.130556345
9	scripting	0.018107652
10	crosssite	0.022634565
11	allows	0.767428160
12	inject	0.022634565
13	arbitrary	0.389475942
14	script	0.022634565
15	html	0.013580739

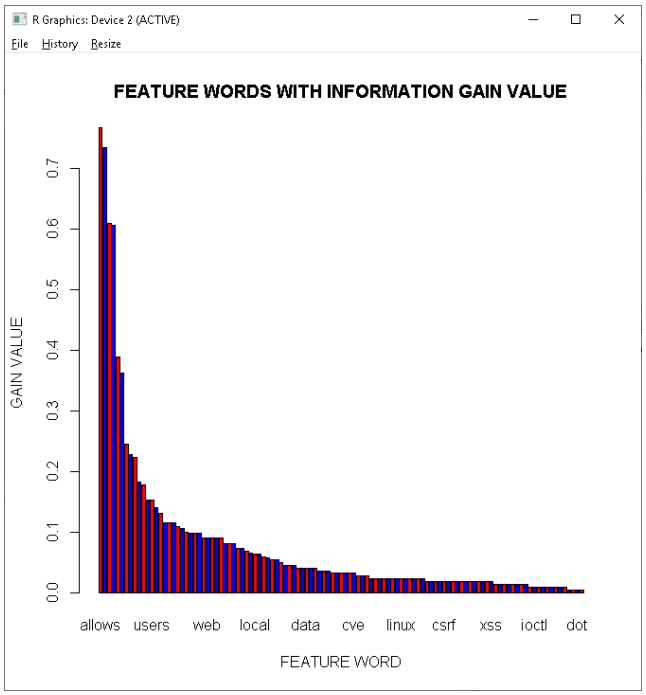

Figure 5.1 Informtion Gain Value for Features (Sorted)

D. OPTIMIZATIONS USING DNN

DNN consists of i) one input layer, ii) multiple hidden layers and iii) one output layer, whose input is feature vector of instance and output is category of instance. It includes mainly two propagation processes, a) forward propagation and b) back propagation. The process of propagation is shown in Algorithm 2.

ALGORITHM 2

INPUT: VECTOR ENCODED DESCRIPTION (If the feature word present in the given description 1 is set otherwise 0 is set. So the vector encoded value contains 113 values i.e., 1 and 0 combinations for 113 terms (feature words). So 113 neurons
OUTPUT: 1 Output neuron with value 0 to 1. The category count is set to 30 so for any input neurons with 113 neuron values, if value between 0 and 0.333 is prepared as output in output neuron, then the category belongs to first (out of 30 categories), 0.334 to 0.666 belongs to second category and so on.

1. Fetch one description and convert to 113 neurons, 10 hidden layer one input weights are set and network is made to run.
2. The output is generated and noted.
3. For all descriptions, the above process is made and output is noted.
4. The weights and biases for hidden and output layer are recalculated for given number of epochs (ten iterations).
5. The forward propagation of the input layer and hidden layer uses 'tanh' as the activation function, while the output layer uses 'softmax' as the activation function.
6. The final weights and bias values are taken for further test description samples.

VI. CONCLUSION

To better analyze as well as manage vulnerabilities according to their belonging classes, improving security performance of the system, and reducing the risk of the system being attacked and corrupted, this study applied DNN (deep neural network) for software vulnerability classification. The analysis of method and construction process of a) TFI and b) DNN are discussed with details. The comparison is made with the vulnerability classification model TFI-DNN to TFI-SVM, TFI-Naïve Bayes and TFI-KNN on the NVD dataset. The results show that the new proposed TFI-DNN model outperforms well to prepare weights and biases. And it is superior to general TF-IDF on comprehensive evaluation indexes. The work in this study showed effectiveness of TFI-DNN during vulnerability classification, and provided a basis for the future research using benchmark vulnerability datasets.

REFERENCES

[1] R. P. Abbott, J. S. Chin, J. E. Donnelley, W. L. Konigsford, S. Tokubo, and D. A. Webb, Security Analysis and Enhancements of Computer Operating Systems. Washington, DC, USA: US Department of Commerce, 1976.

[2] I. R. Bisbey and D. Hollingworth, Protection Analysis: Final Report. Marina Del Rey, CA, USA: Univ. of Southern California, 1978.

[3] A. Gray, “An historical perspective of software vulnerability management,” Inf. Secur. Tech. Rep., vol. 8, no. 4, pp. 34–44, 2003.

[4] P. J. Kim, ‘‘An analytical study on automatic classification of domestic journal articles based on machine learning,’’ J. Korean Soc. Inf. Manage., vol. 35, no. 2, pp. 37–62, 2018.

[5] B. Shua, H. Li, M. Li, Q. Zhang, and C. Tang, “Automatic classification for vulnerability based on machine learning,” in Proc.IEEEInt.Conf.Inf. Automat. (ICIA), Aug. 2013, pp. 312–318.

[6] D. Wijayasekara, M. Manic, and M. McQueen, “Vulnerability identification and classification via text mining bug databases,” in Proc.40th Annu. Conf. IEEE Ind. Electron. Soc., Nov. 2014, pp. 3612–3618.

[7] S. Na, T. Kim, and H. Kim, “A study on the classification of common vulnerabilities and exposures using Naïve Bayes,” in Proc. Int. Conf. Broadband Wireless Comput., Commun. Appl. Cham, Switzerland: Springer, 2016, pp. 657–662.

[8] M. Gawron, F. Cheng, and C. Meinel, “Automatic vulnerability classification using machine learning,” Proc. Int. Conf. Risks Secur. Internet Syst. Cham, Springer, 2017, pp. 3–17.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 248–255.

[10] O. Russakovsky et al., “ImageNet large scale visual recognition challenge,” Int. J. Comput. Vis. vol. 115, no. 3, pp. 211–252, 2015.

[11] W. Xiong et al., “Toward human parity in conversational speech recognition,” IEEE/ACM Trans. Audio Speech Lang. Process., vol. 25, no. 12. PP. 2410–2423, Dec. 2017.

[12] A. Krizhevsky, I. Sutskever, G. E. Hinton, ‘‘ImageNet classification with deep convolutional neural networks,” in Proc. Adv. Neural Inf. Proc. Syst., 2012, pp. 1097–1105.

[13] D. Silveretal., “Mastering the game of Go with deep neural networks and tree search,” Nature vol. 529, pp. 484–489, Jan. 2016.

[14] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daum,”Deep unordered composition rivals syntactic methods for text classification,” in Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process., 2015, pp. 1681–1691.

[15] H. Jo, J.-H. Kim, K.-M. Kim, J.-Ho Chang, J.-H. Eom, and B-T. Zhang, ‘‘Large-scale text classification with deep neural networks,” Comput. Congnit., vol. 23, no. 5, pp. 322–327, 2016.

[16] W. Aziguli et al., ‘‘A robust text classifier based on denoising deep neural network in the analysis of big data,’’ Sci. Program., vol. 2017, 2017, pp. 1–10.