LETTER

Malicious URLs Detection Based on a Novel Optimization Algorithm

Wang BO^{†,††}, Member, Zhang B. FANG[†], Liu X. WEI[†], Zou F. CHENG^{††a)}, and Zhang X. HUA^{†††}, Nonmembers

SUMMARY In this paper, the issue of malicious URL detection is investigated. Firstly a P system is proposed. Then the new P system is introduced to design the optimization algorithm of BP neural network to achieve the malicious URL detection with better performance. In the end some examples are included and corresponding experimental results display the advantage and effectiveness of the optimization algorithm proposed.

key words: URLs detection, optimization, neural network

1. Introduction

Malicious URLs detection has always been the highlight in the research on Web security protection. At present, a malicious URL is mainly detected by black and white list-based URL detection method and machine learning-based URL detection method. According to the first method, a website cannot be visited until confirming the URL is not within the blacklist database through checking the blacklist. Featured by simplicity and efficiency, it has been widely used in many mainstream browsers, such as IE8, Mozilla Firefox 2.0, Safari, and Chrome, etc. However, the method requires regular blacklist maintenance, which results in a high cost and may lead to the problem of judgment omission. These problems largely set restrictions on its popularization. The second method mainly analyzes the problems from many different perspectives such as URL characteristics, domain name characteristics, host characteristics, etc., so as to improve the ability to identify malicious URL. Related researches have received more and more attention during these years.

For instance, in Ref. [1] some typical classification approaches and NLP-based features are adopted to detect phishing URL; in Ref. [2] a linear learning approach of two-stage distance metric and nonlinear Nyström method of kernel approximation are adopted to improve the malicious URLs detection. in Ref. [3] lightweight URL and HTML features are adopted to design the detection method of phishing webpage with better performance; in Ref. [4] a way to detect malicious URL addresses using convolutional neural networks is proposed and can be applied in mobile

devices without significantly affecting its performance; in Ref. [5] a multi-view neural network is introduced and URL feature mining technique is adopted to achieve malware detection.

However, most of these research outcomes are mainly about the extraction and analysis of URL features, and hardly lay emphasis on the research of machine learning. BP neural network (BPNN), as a commonly used machine learning method for URL security testing, was proposed by Rumelhart and Hinton in 1986. As a three layers feedforward network, BP neural network can learn the relationship between input and output without the need of knowing its mathematical expressions in advance. However, “speed” and “accuracy” are the problems that PB neural network needs to notice, and have not been well solved yet.

As a new branch of natural computing [6], membrane computing, proposed by Păun (also called as P system), not only introduces a new distributed parallel information processing technology for computer science, but also provides a new tool for system modeling and simulation [7]–[10]. In view of the above problems, in this paper the P system will be adopted to design the optimization algorithm of BP neural network to achieve URL security detection with better performance. However to our knowledge, the related work has never been carried out at present, and these motivate our investigation.

2. Preliminaries

TF-IDF is a kind of commonly used weighting technique of information exploration.

TF [Term Frequency]:

$$tf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}}$$

where tf_{ij} represents frequency that term t_i appears in text, n_{ij} is the times of term t_i appearing in file d_j , and $\sum_k n_{kj}$ is the total times of all terms appearing in file d_j .

IDF [Inverse Document Frequency]:

$$idf_{ij} = \log \frac{|D|}{|\{j : t_i \in d_j\}|}$$

where idf_{ij} represents the importance of term t_i in the whole file set; $|D|$ represents the total number of files in the corpus; $|\{j : t_i \in d_j\}|$ represents the number of files containing term t_i .

Manuscript received November 20, 2020.

Manuscript publicized January 14, 2021.

[†]The authors are with School of Electrical Engineering and Electronic Information, Xihua University, Chengdu, 610039, China.

^{††}The authors are with School of Electronic Information and Automation, Aba Teachers University, Sichuan, 623002, China.

^{†††}The author is with School of Information, Hiroshima Institute of Technology, Hiroshima-shi, 731–5193 Japan.

a) E-mail: 1724016549@qq.com

DOI: 10.1587/transinf.2020EDL8147

Fig. 1 Framework to produce the sparse matrix from two URLs

TF-IDF:

$$tfidf_{ij} = tf_{ij} \times idf_{ij}$$

Remark 1: It can be seen that the bigger value of $tfidf_{ij}$ means the greater importance this term possesses to the text. Therefore TF-IDF tends to filter out the common terms and retain the important ones.

N-grams refer to N words appearing consecutively in a text. N-grams model is a probabilistic language model based on $N-1$ Markov chain. It infers the structure of sentences by the probability under which N words appear. N-grams text is widely used in text mining and natural language processing tasks.

For instance, URL:

www.foo.com/1

When $N = 3$, one can get corresponding N-gram as:

‘ww.’, ‘w.f’, ‘.fo’, ‘foo’, ‘oo.’, ‘o.c’, ‘.co’, ‘com’, ‘om/’, ‘m/1’

The framework to produce the sparse matrix from a normal URL and a malicious URL is shown in Fig. 1.

Remark 2: $\{(0, 0) 0.2716008248890563\}$ in Fig. 1 means that the TF-IDF value of the first row and the first column of the sparse matrix is 0.2716008248890563.

3. Main Results

The following open dataset is adopted to test our results.

Table 1 Description of Y_{label}

Y_{label}	Sample 1	Sample 2	Sample 3	Sample 4
Class I	1	0	0	1
Class II	0	1	1	0

https://github.com/adamyong-zbf/URL_detection

The total number of URLs in this dataset is 420465, 85% of them are utilized as training samples, and remaining 15% of them are utilized as testing samples.

In view of the characteristic of sparse matrix, “batch size” of BP neural network is set as 2000. Each time 2000 requests are input, and formed into a matrix $X_{train} \in R^{S_0 \times 2000}$ after TF-IDF and N-grams, S_0 is the number of attributes generated by the samples, matrix $Y_{label} \in R^{2 \times 2000}$ represents the classification result of the samples, 1 means that this sample belongs to the class, and 0 means that this sample does not belong to the class. The description of Y_{label} is shown in Table 1.

Remark 3: From Table 1, it can be seen that this Y_{label} contains 4 samples, Samples 1 and 4 belonged to Class I, and Samples 2 and 3 belonged to Class II.

BP neural network: $epoch$ is the training period, lr is the learning rate, S_0 and S_2 are the neuron numbers of input and output layers, S_1 is the neuron number of hidden layer, such that

$$S_1 = \sqrt{S_0 + S_2} + a$$

where

$$a \in [1, 10]$$

In this paper, following fitness function is adopted as:

$$Fitness = \frac{1}{SE}$$

where

$$SE = \text{sumsq}(X_{test} - A_2)$$

$$A_2 = \text{purelin}(W_2 A_1, b_2)$$

$$A_1 = \text{tansig}(W_1 X_{train}, b_1)$$

$$W_1 = X_{out}(1, \dots, S_0 S_1)$$

$$W_2 = X_{out}(S_0 S_1 + 1, \dots, S_0 S_1 + S_1 S_2)$$

$$b_1 = X_{out}(S_0 S_1 + S_1 S_2 + 1, \dots, S_0 S_1 + S_1 S_2 + S_1)$$

$$b_2 = X_{out}(S_0 S_1 + S_1 S_2 + S_1 + 1, \dots, S_0 S_1 + S_1 S_2 + S_1 + S_2)$$

$$S = R S_1 + S_2 S_1 + S_1 + S_2$$

where sumsq denotes the sum of squares operator, purelin denotes the linear transformation, tansig denotes the sigmoid transformation, X_{train} denotes the training data, X_{test} denotes the testing data.

Adjustable parameters: W_1 denotes the first weight matrix, W_2 denotes the second weight matrix, b_1 denotes the first offset value, b_2 denotes the second offset value.

PSO:

$$v_{id} = w v_{id} + c_1 \text{random}(0, 1)(p_{id} - x_{id})$$

Fig. 2 Framework of a new P system

$$+ c_2 \text{random}(0, 1)(p_{gd} - x_{id})$$

$$x_{id} = x_{id} + v_{id}$$

where x_{id} , $1 \leq i \leq M$, $1 \leq d \leq D$, M denotes swarm size, D denotes particle dimension, c_1 , c_2 are learning factors, w is inertia weight (IW), and has a great influence on the optimal performance of PSO.

Remark 4: Large inertia weight (IW) means better searching precision, and more particles will move towards the optimal point just be found, and may result in premature problem if the point just found is the local optimal. Conversely, small inertia weight (IW) means faster convergence speed and larger probability of jumping off local optimal point, and may result in big oscillation and precision problem.

To solve this problem, membrane computing is introduced to design the optimization algorithm with better performance. The framework of a P system is proposed as Fig. 2.

There are four membranes in the given P system. In Membs. 1 and 2, PSO has the large IW. In Membs. 3 and 4, PSO has the small IW. The interaction rules are as follows:

The particle with the best fitness of Memb. 4 is transferred and replaces the particle with worst fitness of Memb. 3;

The particle with the best fitness of Memb. 3 is transferred and replaces the particle with the worst fitness of Memb. 1;

The particle with the best fitness of Memb. 2 is transferred and replaces the particle with the second-worst fitness of Memb. 1;

The particle with the best fitness of Memb. 1 is transferred and replaces the particle with the worst fitness of Membs. 2 and 4.

When the iteration is over, the particle with the best fitness of Memb. 1 is output as the ultimate optimum.

Remark 5: From Fig. 2 it can be seen that there are 4 membranes in the given P system. In Membs. 3 and 4, PSO with small IW is utilized to carry on the global search to increase the convergence speed. In Membs. 1 and 2, PSO with large IW is utilized to carry on the local search to improve the searching precision and decrease the oscillation.

“Accuracy” represents the correct classification rate, “Precision” represents the classification exactness, “Recall” represents the classification completeness. They can be used to evaluate the classification performance of algorithm, and the definitions are as follow:

Table 2 Test results of classification performance of algorithms

	A	P	R
PSO-BP	0.9532	0.9651	0.9426
Proposed	0.9675	0.9815	0.9548

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where

TP denotes True Positive

TN denotes True Negative

FP denotes False Positive

FN denotes False Negative

Let $lr = 0.01$, $epoch = 1000$, $T_{max} = 100$, $a = 3$, $S_1 = 163$, $S_2 = 2$, $w_1 = 0.4$, $w_2 = 0.9$, $c_1 = 2$, $c_2 = 2$, $M = 50$, conduct 10 times experiments, and the average of test results are displayed in Table 2.

Remark 6: In Table 2, A denotes “Accuracy”, P denotes “Precision” and R denotes “Recall”, which are important classification performance indexes. For them, the higher value means the better classification performance. From Table 2, it can be seen that A, P and R of the proposed algorithm are 0.9675, 0.9815 and 0.9548 respectively, larger than 0.9532, 0.9651 and 0.9426 of BP algorithm respectively, which means that proposed method has better classification performance.

4. Conclusion

The paper discussed the issue of malicious URL detection. Based on a new P system constructed and BP neural network, a new optimization algorithm has been proposed to achieve URL security detection with better performance. Finally some experiments have been carried out to display the advantage and effectiveness of the optimization algorithm.

Acknowledgements

Authors would like to thank the editor and anonymous reviewers for their insightful and constructive comments and suggestions. The work is supported by the Traction Power State Key Laboratory Open Issue (TPL1907), and the Sichuan Science and Technology Program under Grant (2020YFG0184).

References

- [1] O.K. Sahingoz, E. Buber, O. Demir, and B. Diri, “Machine learning based phishing detection from URLs,” *Expert Systems with Applications*, vol.1171, pp.345–357, 2019.
- [2] T. Li, G. Kou, and Y. Peng, “Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods,” *Information Systems*, vol.91, p.101494, 2020.
- [3] Y. Li, Z. Yang, X. Chen, H. Yuan, and W. Liu, “A stacking model using URL and HTML features for phishing webpage detection,”

- Future Generation Computer Systems, vol.94, pp.27–39, 2019.
- [4] W. Wei, Q. Ke, J. Nowak, M. Korytkowski, and M. Woźniak, “Accurate and fast URL phishing detector: A convolutional neural network approach,” *Computer Networks*, vol.1784, 107275, 2020.
- [5] S. Wang, Z. Chen, Q. Yan, K. Ji, and M. Conti, “Deep and broad URL feature mining for android malware detection,” *Information Sciences*, vol.513, pp.600–613, 2020.
- [6] G. Păun, “Membrane computing,” *Lecture Notes in Computer Science*, vol.2751, pp.284–295, 2003.
- [7] L. Pan, X. Zeng, and X. Zhang, “Time-free spiking neural P systems,” *Neural Computation*, vol.23, no.5, pp.1320–1342, 2011.
- [8] G. Singh and K. Deep, “A new membrane algorithm using the rules of particle swarm optimization incorporated within the framework of cell-like p-systems to solve Sudoku,” *Appl. Soft Comput*, vol.45, pp.27–39, 2016.
- [9] J. Xiao, X. Zhang, and J. Xu, “A membrane evolutionary algorithm for DNA sequence design in DNA computing,” *Chin. Sci. Bull.*, vol.57, pp.698–706, 2012.
- [10] G. Zhang, J. Cheng, and M. Gheorghe, “A hybrid approach based on differential evolution and tissue membrane systems for solving constrained manufacturing parameter optimization problems,” *Appl. Soft Comput.*, vol.13, pp.1528–1542, 2013.
-