Research on phishing webpage detection technology based on CNN-BiLSTM algorithm

Qiao Zhang¹, Youjun Bu², Bo Chen², Sureng Zhang² and Xiangyu Lu²

¹School of software, Zhengzhou University, Zhengzhou, Henan, 450000, China
²Information Technology Institute, PLA Strategic Support Force Information Engineering University, Zhengzhou, Henan, 450000, China

*Corresponding author’s e-mail: stulea@gs.zzu.edu.cn

Abstract. The rapid development of the Internet has also brought opportunities for some illegal elements. Network attackers steal sensitive information from victims through phishing webpages to obtain economic benefits. Currently, the commonly used detection methods for phishing webpages, based on blacklist detection and webpage content feature detection, have the problems of being unable to detect newly emerging phishing webpages or requiring manual extraction of webpage features. Therefore, researchers have used Convolution Neural Network (CNN) to detect phishing webpages by automatically extracting URL features. However, its method has some limitations: (1) The memory is limited when the URL is transformed into the feature matrix, and the embedding vector of new words cannot be obtained or the effective information of sensitive words is lost; (2) the long-distance dependent feature of the URL cannot be obtained. In response to the above challenges, we proposes a phishing detection method based on CNN and Bi-directional Long Short-Term Memory (Bi-LSTM) based on existing work: based on sensitive word segmentation-- comprehensively using two existing URL segmentation methods before converting URL into eigenvector matrix; adding Bi-LSTM on the basis of convolutional neural network to obtain URL long-distance dependent features. Experimental results show that this method can achieve high accuracy, recall rate and F1 value.

1. Introduction
In recent years, with the rapid development of the Internet, applications based on the Internet, such as online shopping, e-commerce and social networking, have brought great convenience to people's work, life and entertainment. Therefore, more and more people begin to contact and use the Internet. According to the statistics of CNNIC⁴, by March 2020, the number of Internet users in China reached 904 million, and the Internet penetration rate reached 64.5%. At the same time, the information of Internet users is also facing security threats, such as phishing to steal personal sensitive information and obtain economic benefits. By June 2020, the number of phishing websites identified by China Anti phishing alliance has reached 469252⁵. The number of phishing websites is huge, which has brought serious harm to the public. Therefore, how to detect phishing websites timely and effectively has become an urgent problem.

2. Related work
For phishing webpages, there are currently three main types of detection methods: blacklist-based detection, webpage content feature detection, and URL feature detection.
The blacklist detection only performs simple database query operations, and the detection rate is fast, simple and convenient, but its limitation is to continuously collect phishing website samples and update the blacklist database in time. Content-based detection methods first need to obtain web content, and then judge the legitimacy of the webpage to be tested based on the similarity of web content or machine learning technology. This method needs to obtain web content, which increases the risk of the client. In addition, it requires a lot of manual feature engineering. Many of the features need to be confirmed by relevant experts. Its performance depends heavily on the quality of the manually extracted features. The detection model is easily bypassed by phishing attackers due to relatively fixed characteristics.

Based on URL feature detection, current existing methods mainly use neural networks to automatically extract URL features to determine the legitimacy of web pages, as shown in Figure 1.

A URL is essentially a series of characters or words separated by special characters. Converting the URL into a feature vector that the neural network can recognize is to get its matrix representation $u \rightarrow X \in R^{i \times k}$, matrix contains a set of adjacent components $X_i (i = 1, 2, 3, ..., L)$, where $X_i$ is the vector representation of the characters or words of the URL, and $X_i \in R^k$ is a k-dimensional vector. Reference [3] divides the URLs in the dataset at word level according to special characters such as "/", "@", "&", ".", etc. and forms a corpus. Each word in the training corpus is represented as a vector, and then the URL is segmented to obtain the vector representation of the word and combined to form a vector matrix, which is input into the convolution neural network to determine the type of the corresponding URL. Reference [4-8] divides URLs by characters, obtains vectors of each character and combines them to form a vector matrix, and then inputs the matrix into convolution neural network to determine the corresponding URL type.

In summary, the method of detecting phishing webpages based on URL features mainly uses a single neural network and two word segmentation methods. The limitations of this method are as follows: Segmentation of URLs based on words, using special characters to segment URLs may make the number of words quite large, resulting in a proportional increase in the characteristics of the dataset, and the embedding vectors of newly appearing words cannot be obtained during testing; Dividing the URL based on characters will cause some sensitive words such as "login", "password", "registed", etc. to lose some valid information; Using a single neural network such as CNN only obtains the local features of the URL, and cannot obtain the sequence features of the URL.

Aiming at the problems of the above methods, we propose a method based on sensitive word segmentation. According to the special characters, the URL is divided in word level, and the special characters are treated as words to obtain the effective information of the special characters, and then the non-sensitive words are divided in the character level, and the sensitive words are regarded as a whole with the rest of the characters to distinguish, and this can clearly mark the key information in the URL, which is beneficial to the neural network classifier to extract more representative features. Neural network classifier uses convolutional neural network and bidirectional long short memory network to extract more abundant features from URL.

3. Model configuration
The model framework for detecting URL categories based on the hybrid model CNN-BiLSTM proposed in this paper is shown in Figure 2. The specific process and detection model parameter configuration are as follows:
Step1. Segment URL based on sensitive word segmentation method.

Step2. According to the URL data set and sensitive vocabulary (table 1), the total length L of characters and keywords in each URL is determined to be 300. If the URL length is more than 300, the extra characters will be truncated at the end of the URL. If the URL length is less than 300, the < pad > tag is used as an additional word at the end of the URL. If an unknown character appears in the URL, it is indicated by the unknown character mark < unk >.

Step3. By analyzing the URL data set and sensitive vocabulary, the total number of different characters and sensitive words is 121. Build a mapping table to assign unique codes to characters and sensitive words, as shown in table 2.

Step4. The digital coding of URL is transformed into two-dimensional dense feature vector matrix by word embedding matrix. Firstly, the URL is transformed into a 300 * 1 matrix $X$ according to the mapping table of characters and sensitive words, as shown in formula 1 where $x_i$ is a one-dimensional column vector $i = 1, 2, 3, ..., 300$. Then, the matrix $X$ is transformed into a 300 * 32 two-dimensional dense matrix containing semantic information through the embedding layer of neural network, as shown in formula 2, where $\hat{x}_i$ is a 32 dimensional column vector.

$$X = (x_1, x_2, ..., x_{300})$$ (1)

$$X' = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_{300})$$ (2)

Step5. The feature vector matrix is input into the convolution neural network, and the local features are automatically extracted from the feature matrix by the convolution kernel. The height of the convolution kernel is set to 2, the width is consistent with the dimension of the character vector is 32, the number of convolution kernels is 200, and the sliding step size of convolution kernel is set to 1. For a convolution kernel, the URL embedding matrix obtained at the ith sliding window is set as shown in formula 3, where $\hat{x}_i$ is the vector representation of characters or sensitive words. The new feature $c'_{i}$ generated by convolution operation is set as shown in formula 4, where $W_f$ and $b_f$ is the weight matrix and bias term, and $\sigma$ is the activation function relu. The convolution kernel traverses the entire embedding matrix to generate a feature map $c'_f$, which is recorded as the formula 5.

$$X'_i = [\hat{x}_i, \hat{x}_{i+1}, ..., \hat{x}_{i+h-1}]$$ (3)

$$c'_i = \sigma(W_f \cdot X'_i + b_f)$$ (4)

$$c' = [c'_1, c'_2, ..., c'_{300-h+1}]$$ (5)

Step6. Maximize the pooling window of $c'$ to obtain more representative features (pooling window size is 2, pooling step size is 1). Set at the i-th pooling window, the new feature map $m'_i$ after pooling is as shown in formula 6, then the pooling window traverses the entire $c'$ and the new feature map is obtained as shown in formula 7.

$$m'_i = \max(c'_i, c'_{i+1}, ..., c'_{i+pl-1})$$ (6)

$$m'' = \max(m'_1, m'_2, ..., m'_{(300-h+1)/pl})$$ (7)

Step7. Stack the new feature maps obtained after $X'$ is pooled by all convolution kernels to obtain a sequence matrix as shown in formula 8, where $s = \lceil (L - h + 1)/pl \rceil$, $m_p$ is the feature vector composed of all convolution check URL words embedded in the same region of the matrix after convolution and pooling operations, $m_p \in R^n$, and $n$ is the number of convolution kernels.

$$MP = [m_p, m_p, ..., m_p]$$ (8)
Step8. Consider $M^p$ as the sequence information on the time axis as the input of BiLSTM, and $m^p_i$ corresponds to the input of BiLSTM at the $i$-th moment. The forward LSTM memorizes the information before the time $i = s$ through the forget gate, input gate, and output gate. The output at this moment is recorded as $h^F_s$. The reverse LSTM remembers the information after $i = s$ through three gates: forget gate, input gate, and output gate. The output at this moment is recorded as $h^R_s$, and the last moment output of LSTM in two different directions is spliced as $h = h^F_s \oplus h^R_s$ to obtain long-distance dependent features in different directions of the URL.

Step9. Set the number of neurons in the fully connected layer to 2, and calculate the probability that the URL to be tested belongs to a phishing or legitimate webpage through the activation function softmax, as shown in formula 9, where $z_i = w_i h + b_i$, $w_i$ and $b_i$ are the weight and bias parameters, and $i$ are the URL category index (0 means phishing URL, 1 means legal URL), $k$ is the total number of URL categories, the value is 2.

$$p_i = e^{z_i} / \sum_{i=1}^{k} e^{z_i}$$

Table 1. Sensitive word.

| Sensitive words | account admin administrator auth bank client confirm cmd email host login password pay private registered safe secure security sign service signin submit user update validation verification webscr |

Fig.2 model framework for detecting URL categories based on the hybrid model CNN-BiLSTM.
Table 2. Character and sensitive word mapping table.

| Character Code | Code |
|---------------|------|
| abcdefghijklmnopqrstuvwxyz | 1-26 |
| ABCDEFGHIJKLMNOPQRSTUVWXYZ | 27-52 |
| 0123456789 | 53-62 |
| .,=/+\&:;'?><{}()[]|~!@#$%* | 63-92 |
| account, admin, administrator, auth, bank, client, confirm, cmd, email, host, login, password, pay, private, regist, safe, secure, security, sign, service, signin, submit, user, update, validation, verification, webscr | 93-119 |
| Fill character mark <PAD> | 120 |
| Unknown character mark <UNK> | 121 |

4. Experiment and analysis

4.1. Experimental data

The data set used in this article includes open source samples provided by multiple platforms, obtaining phishing URLs from PhishTank and MalwarePatrol, and legitimate URLs from DMOZ and Alexa to enrich the source of URL data. PhishThank is an anti-phishing website where users can submit, verify and share phishing data. MalwarePatrol is similar to PhishTank, where users can download phishing URLs. DMOZ is the largest global directory community maintained and built by volunteers from all over the world. It aims to include excellent websites through which legal URL data sets can be obtained. Alexa is a website that specializes in publishing website rankings in the world. It currently has a large number of URLs and detailed website ranking information. The top-ranked websites are collected as a legal URL data set. After deduplicating the data, the data set contains a total of 206,200 tagged URL samples, of which 105,100 are phishing samples and 101,100 are legitimate samples. The ratio of the two is approximately 1:1. Table 3 shows an example of a partial URL sample.

| Sample source | URL |
|---------------|-----|
| PhishTank | https://www.faresproducts.com/wp-admin/includes/ionos-GE/#info@koalanin
https://rakuten.co.jp-rktqlhxtgshamexy.gbyhmisif.work/?signin=a&email=a |
| MalwarePatrol | https://demcorknitwear.com/32787231/index.php?email=aaaa@example.jp
https://secure-recovery.xyz/m.checkpoint.htm |
| DMOZ | http://www.mjzjswh.gov.cn
http://www.myzte.cn |
| Alexa | http://www.tmall.com
http://www.babyytree.com |

4.2. Evaluation criteria

In order to verify the effectiveness of phishing webpage detection method, accuracy, precision, recall and F1 value are used as evaluation indexes. The calculation formula is shown in 10-13, where TP (true positive) represents the actual number of phishing web pages predicted, FP (false positive) represents the actual number of legitimate Web pages predicted, TN represents the actual number of legitimate Web pages predicted, and FN (false negative) represents the actual number of predicted legitimate Web pages. Precision represents the proportion of the web pages correctly judged as phishing web pages in all the web pages judged as phishing web pages, which reflects the distinguishing ability of detection methods for legitimate Web pages, while recall reflects the recognition ability of phishing web pages. F1 value takes both accuracy and accuracy into account,
which is the weighted average of the two, and can comprehensively evaluate the performance of the detection model.

\[
\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}
\]

\[
\text{Precision} = \frac{TP}{(TP + FP)}
\]

\[
\text{Recall} = \frac{TP}{(TP + FN)}
\]

\[
F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

4.3. Experimental result

We use a 10-fold cross-validation method on the URL data set, which means that the samples are divided into 10 groups, each of which contains 10,510 phishing URLs and 10110 legal URLs as the test set, and the other 9 groups contain 94590 phishing URLs and 90990 legal URLs as the test set. The process circulates 10 times to ensure that each group of sample data can be predicted as test set, and the average value of the 10 test results obtained is used to evaluate the detection ability of the model. As shown in Figure 3, the average change curve of the accuracy rate of the proposed model in this paper on the training set and test set under 10 fold cross validation. It can be seen from the figure that the parameters of the model converge normally during the training process. When the number of training rounds is 30, the training and testing accuracy of the model tends to be stable.

In addition, in order to reflect the advantages of the detection model proposed in this paper, it is compared with the character-level CNN detection model char_CNN that divides url by characters, the word-level CNN detection model word_CNN that divides url by words, the SW_CNN that divides url by sensitive words (SW) and the hybrid detection model SW_CNN_RNN and SW_CNN_LSTM that divide URLs by SW. The same part of the model structure uses the same parameters. Under the same experimental environment and data set, the detection results of the six models are shown in table 4. Meanwhile, the accuracy of these models in the training set and verification set is recorded, as shown in Fig 4 and Fig 5. Combined with table 4, fig 4 and fig 5, it can be seen that the six detection models in this paper have achieved high detection accuracy on the same data set.

Among them, the detection model char_CNN has reached a high accuracy rate at the beginning of the training set and verification set, but as the number of training rounds increases, the accuracy rate is not improved much. The accuracy change curve of word_CNN on the training set and validation set is similar to that of char_CNN, but the accuracy rate is lower than that of the char_CNN model. This result may be derived from the following three aspects: the effective information of the special characters is ignored when the URL is segmented by special characters such as ".", ",", and "?", in order to avoid memory constraints, the words that appear only once in the dataset are marked as <unk>; Unable to obtain valid information about newly appeared words. SW_CNN is more accurate than char_CNN because it can obtain effective information of sensitive words in URL. Although SW_CNN_RNN uses a hybrid network model to extract URL features, because RNN cannot obtain long-distance dependent features of URLs, its
detection accuracy is lower than that of char_CNN and SW_CNN with a single model structure. SW_CNN_LSTM has a low accuracy rate at the beginning of training, but as the number of training rounds increases, the accuracy rate has been rapidly improved. When the model basically converges, its accuracy on the training set and test set is higher than SW_CNN. Compared with the above models, the detection model SW_CNN_BiLSTM proposed in this paper can obtain more sufficient URL features and achieve the highest detection accuracy, precision, recall, and F1 value.

5. Conclusions and future work
In order to solve the problems of existing phishing web page detection methods, such as manual feature extraction, unable to identify new phishing web pages or insufficient feature extraction, we propose a phishing web page detection technology based on CNN-BiLSTM algorithm, which can automatically extract URL features by hybrid neural network to detect new phishing web pages. This method first performs word segmentation processing on URL based on sensitive word segmentation, then converts it into a feature vector matrix, automatically extracts its local features through CNN, acquires its bidirectional long-distance dependent features through BiLSTM. The multi-level features are input into the full connection layer and classified by the activation function softmax. Compared with character level CNN, word level CNN and other phishing webpage detection technologies, the results show that the proposed phishing web page detection technology based on CNN-BiLSTM achieves high results in accuracy, recall rate and F1 value. Future research work will use the adversarial generative network to generate phishing URLs as input to analyze the robustness of the proposed model.

Acknowledgments
This research is partially supported by the National Key Technology Research and Development Program of China(2017YFB0803201, 2017YFB0803204), National Natural Science Foundation of China (61572519, 61802429, 61521003), Scientific Research Project of Shanghai Science and Technology Commission(16DZ1120503) and China Postdoctoral Fund Project (44595)
References

[1] China Internet Information Center. (2020) The 45th statistical report on China's Internet development. http://apac.cn/gzdt/202007/P020200716563107001351.pdf.

[2] China Anti phishing website alliance. (2020) Brief report on the handling of phishing websites in June 2020 [EB/OL] http://apac.cn/gzdt/202007/P020200716563107001351.pdf.

[3] Zhang M, Xu B, Bai S, et al. (2017) A Deep Learning Method to Detect Web Attacks Using a Specially Designed CNN In: Neural Information Processing. Springer, pp. 828-836.

[4] Joshua S, Konstantin B. (2017) A Character-Level Convolutional Neural Network with Embeddings For Detecting Malicious URLs, File Paths and Registry Keys. arXiv preprint arXiv:1702.08568.

[5] Cui Y P, Liu M, Hu J W. (2020) Malicious web request detection technology based on CNN. Computer science, 047(2): 281-286.

[6] Jiahong W, Zhenguo Y, L, et al. (2019) Convolutional Neural Network with Character Embeddings for Malicious Web Request Detection. In: IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking. Xiamen China, pp. 622-627.

[7] A. Vazhayil, R. Vinayakumar and K. P. Soman, (2018) Comparative Study of the Detection of Malicious URLs Using Shallow and Deep Networks. In: 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bangalore, pp. 1-6.