IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON DATA MINING FOR INTERNET OF THINGS

Received May 3, 2020, accepted May 11, 2020, date of publication May 18, 2020, date of current version June 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2995157

CNN Based Malicious Website Detection by Invalidating Multiple Web Spams

DONGJIE LIU[1,2] AND JONG-HYOUK LEE[3], (Senior Member, IEEE)
[1]Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China
[2]School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100190, China
[3]Department of Computer and Information Security, Sejong University, Seoul 13557, South Korea

Corresponding author: Jong-Hyouk Lee (jonghyouk@sejong.ac.kr)

This work was supported by the Ministry of Science and ICT (MSIT), South Korea, through the Information Technology Research Center (ITRC) Support Program supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP) under Grant IITP-2020-2015-0-00403.

ABSTRACT Although a variety of techniques to detect malicious websites have been proposed, it becomes more and more difficult for those methods to provide a satisfying result nowadays. Many malicious websites can still escape detection with various Web spam techniques. In this paper, we first summarize three types of Web spam techniques used by malicious websites, such as redirection spam, hidden IFrame spam, and content hiding spam. We then present a new detection method that adopts the perspective of users and takes screenshots of malicious webpages to invalidate Web spams. The proposed detection method uses a Convolutional Neural Network, which is a class of deep neural networks, as a classification algorithm. In order to verify the effectiveness of the method, two different experiments have been conducted. First, the proposed method was tested based on a constructed complex dataset. We present comparison results between the proposed method and representative machine learning-based detection algorithms. Second, the proposed method was tested to detect malicious websites in a real-world Web environment for three months. These experimental results illustrate that the proposed method has a better performance and is applicable to a practical Web environment.

INDEX TERMS Convolutional neural network, machine learning, malicious website detection.

I. INTRODUCTION

The Internet has become an indispensable part of people's life. However, while the Internet brings prosperity, it is also causing problems like illegal websites, fake medical websites, pornographic, gambling, etc. Despite the fact that various detection techniques were applied, the number of malicious websites continues to grow. The large amount of malicious information on the Internet is harmful to the health of Internet users, especially kids and teens [1], [2].

To detect malicious websites, researchers have come up with a lot of methods, including heuristic methods, machine learning based methods, and so on. Nowadays people usually use machine learning methods to analyze text and image information from websites but due to the huge temptation of profits, the malicious websites use a variety of Internet spam techniques to evade regulation. Therefore, to solve the problem, it is necessary to first summarize the commonly used spam techniques for preventing malicious website detection methods. For instance, the redirection spam and hidden IFrame spam provide false content to crawler, which leads to severe false negatives. The content hiding spam usually presents legitimate content but contains invisible malicious information, which cannot be seen by browsers and users so that it leads to a high false-positive rate to the detection methods.

A redirection spam gives the crawler a Web page containing the wrong content, but automatically displays an unrelated page to users. JavaScript redirection is the best Web spam redirection technique, which renders spam content to a script-independent crawler but automatically redirects a browser that can parse the script to a different URL when the page is loaded [3]. For example, http://www.bowas.cn/ is a typical malicious site using the redirection spam technique. From the script-agnostic perspective, it provides entertainment and gossip information; but for a scripted

The associate editor coordinating the review of this manuscript and approving it for publication was Chun-Wei Tsai.

FIGURE 1. Source code fragment of http://www.qpbuh.cn.

browser, the content presented on the Web page is not the case. For instance, as soon as the source code is loaded, the browser will execute a JavaScript file with the path http://www.tuniubb.com/js1/tz.js, which contains a redirection operation to the URL: http://www.tuniubb.com/js1/11.php. Then the browser will load this PHP page, but 11.php is not the end. It further goes through 302 redirects to http://www.wuxin4.com/. In other words, what users eventually see in the browser is http://www.wuxin4.com/, which is a porn page. It is seen that the loading of http://www.bowas.cn/ experiences two different redirections. The first is executing the content of the JavaScript file, and the second is executing a 302 redirection. The redirection process is transparent to the users, who finally access a pornographic website, and thus the purpose of criminals is achieved.

An IFrame is a <iframe> tag used to nest other pages in a page. The IFrame is often used to nest slow-loading third-party content such as ICONS and advertisements. It is a convenient function as people only need to modify <iframe> content, and then the nested web content are modified consistently. However, this advantage is also put to bad use by malicious websites to avoid regulation. For example, the malicious website http://www.qpbuh.cn/ adopts the hidden IFrame spam technique. For a script-agnostic parser, the page is a legal news website. But for users, it is a gambling website, which is presented by https://www.66youyi.com/. Fig. 1 shows a source code fragment of this website. Line 27 shows the website embeds an IFrame with JavaScript. The embedded uniform resource locator (URL) is presented by string concatenation instead of a single string to confuse the parser.

Unlike the redirection spam and hidden IFrame spam, the content hiding spam (including text and hyperlink hiding spam) provides a same page to the crawler and browser, but the browser only shows part of the content. In other words, some text (including the anchor text) is invisible to users. Geng, *et al.* give a detailed taxonomy of hyperlink hiding spam, which is one typical form of content hiding spam [4]. The statistics show that many legitimate websites contain hidden anchors and hyperlinks. But unfortunately, most of the hidden anchors also consist of malicious terms. As a result, a legitimate website with hidden links and texts is easily misjudged as a malicious website because it contains a large amount of malicious vocabulary, although the vocabulary is invisible to users.

The wide use of website design techniques, especially JavaScript, makes malicious redirection, IFrame and, content hiding much easier to use. At the same time it makes malicious website detection more difficult. One biggest challenge malicious website detection now faces is that it is hard to determine whether the parsed web content is correct for analysis or not. It leads to a low performance of the malicious website detection methods.

However, although the difficulty seems hard to overcome, there is still one way to avoid all the confusions caused by the spam techniques and keep away from the complex resolving process. The way is to judge a website by what the website finally presents to users. In other words, if the users finally see fake medicine advertisements, pornography, or other illegal contents, it is a malicious website.

Therefore, according to the analysis above, we adopt the end-users' view and propose a malicious website detection method using a Convolutional Neural Network (CNN) to learn and recognize screenshot images of webpages. In order to eliminate the influence of chaotic and complex information, our method realizes the independence of the used-to-be necessary information provided by source code of parsed webpages. What is more, it makes use of the good performance of CNNs in image learning and classification. The proposed method can reduce the false positive rate and increase the coverage rate so that it can effectively prevent the above mentioned three spam techniques.

The rest of this paper, which is an extension of the paper presented in [5], is organized as follows: Section II provides some related work. Section III presents our proposed malicious website detection method. Section IV presents the conducted experimental results. Section V provides some discussions. Section VI concludes this paper.

II. RELATED WORK

With the accelerating development of Internet technology, malicious websites are widely spread and related techniques keep innovating as well. Accordingly, malicious websites detection methods are also updating and improving. Early

Internet filtering software rely more on non-machine learning based methods, such as blacklisting and sandbox [6]–[8]. As machine learning techniques are advanced [9]–[11], most detection methods start using machine learning based techniques, such as AdaBoost, Naive Bayes, Random Forest, Support Vector Machines (SVM), and so on [12]–[18]. At present, deep learning methods like a CNN and a Recurrent Neural Network (RNN) are becoming more popular and have shown good performance in malicious websites detection [19]–[25].

A. NON-MACHINE LEARNING TECHNIQUES

Early detection methods are not based on machine learning techniques. Fukushima *et al.* analyzed the domain information of malicious websites and evaluated their reputations of Internet Protocol (IP) address blocks and registrars, and based on these they proposed a blacklisting method that combines IP address blocks and registrars with low reputation [6]. Dewald *et al.* proposed ADSandbox on the users' side to detect JavaScript based attacks and they achieved a false positive rate of zero [7]. Zhang *et al.* introduced a blacklisting system based on a ranking scheme, which uses the attacker's history and the attacker's most recent log generation pattern to measure the close relationship between the source of the attack and the contributor. The blacklist system also integrates a number of log prefilters and severity measures to capture how well an attacker's alert pattern matches the spread of common malware [8].

These old fashioned methods usually cannot achieve an overall satisfying accuracy but they can efficiently select a suspicious URL set. This feature makes the old methods useful as a preprocessing step for methods of higher accuracy.

B. CLASSICAL MACHINE LEARNING TECHNIQUES

Researchers extract various types of information and adopt different classical machine learning algorithms for classification. Some researches mainly use text features for classification. Ankit and Gupta extracted features from URLs and source code from users' side and adopted Random Forest as the classifier [18]. Desai *et al.* selected 22 features such as URL length and Google Index and adopted SVM, K Nearest-Neighbor (KNN) and Random Forest as the classifier [16]. Gupta and Sachdeva used SVM to detect malicious URLs in Facebook [17]. Altay *et al.* extracted keywords from the requested HTML contents of the webpages and used SVM, maximum entropy, and extreme learning machines for the classification of websites [12]. In addition to text information, image information are also used for detection. Geng *et al.* analyzed visual brand entities like favicon, logo, and copyright notice in phishing Web sites, extracted brand features and brand authorization features and used C4.5 as the classification algorithm [13]. Marchal *et al.* captured the images of webpages and used them as the phishing detection data [14], but they only extracted information on terms from the images through optical character recognition [15], and left over other potentially useful information.

These researches extracted various information for classification, such as URL related information, text, and image information from html content but none of them have considered multiple Web spam techniques widely used in malicious websites, which will cause inefficient detection. As discussed before, the redirection spam and hidden IFrame spam provide false content to a crawler, which leads to severe false negatives and the content hiding spam contains legitimate content but with invisible sensitive words, which cannot be seen by browsers and users, but not to detection model, which leads to high false-positive rate. In other words, different types of spam techniques make parsed html contents much less meaningful, and thus yield lower detection accuracy.

C. DEEP LEARNING TECHNIQUES

Nowadays more researchers start to adopt deep learning methods to detect malicious websites. Some research compares classical machine learning methods with deep learning techniques. Sirageldin *et al.* selected URL lexical features and page-contents features and used SVM, Decision Tree, Naive Bayes, KNN and Artificial Neural Network (ANN) as the classification algorithms and ANN performs best [21]. Harikrishnan *et al.* used random split and time split to split the data for training, and adopts both classical machine learning techniques (e.g., SVM, Logistic regression, Naive Bayes, KNN, Decision Tree, Random Forest, AdaBoost) and deep learning techniques (e.g., CNN, Long Short-Term Memory models (LSTM), and CNN-LSTM) [23]. Braşoveanu and Andonie also adopted both classical and deep learning techniques such as Logistic Regression, Decision Tree, SVM, CNN, and Bidirectional LSTM (BiLSTM) [19]. Generally deep learning methods perform better than traditional ones. However, in their data sets, the numbers of malicious and legitimate URLs are balanced, which is not what happens in a real Internet world. In the real Internet world, unbalanced positive and negative data actually cause the detection difficulty high and affect the detection accuracy. Besides, the spam techniques which will weaken detection performance are not taken into consideration in their work.

Some recent works focus on deep learning methods. For example, Peng *et al.* proposed a joint CNN and LSTM based attention mechanism [25]; Mahudeswaran and Liu combined RNN and LSTM [20] and Wang *et al.* put forward a bidirectional LSTM algorithm based on CNN and independent RNN [22]. These research proved their method best in their data sets. However, the number of legitimate and malicious data in their experiments are still similar and they don't take spam techniques into account either. Smadi *et al.* combined neural network with reinforcement learning to detect phishing in the online mode [24], which adapts to the real Internet much better but still without spam techniques considered.

Therefore, due to the limited, not-supposed-to-be balanced data which are mainly gained from the source code and URL, and most important, spam techniques like redirection spam, hidden IFrame spam, and content (text/hyperlink) hidden spam, a severe detection evasion still remains unsolved.

Zhou *et al.* took a cloaking spam into consideration and extracted the text and structure features of HTML code, which was then parsed through simulating the progress how browsers work [26]. This method reduces the false negative rate but some legitimate websites can be sometimes misjudged due to the legitimate websites can also contain hidden text or hyperlinks. Besides, this method only analyzes text information and excludes image information, which may also miss some malicious websites using pictures to avoid detection. At last, a JavaScript parser is not capable enough to handle the large amount of websites.

On this basis, we adopt the perspective of users and takes screenshots of malicious webpages to invalidate Web spams and uses a CNN as our classification algorithm. Experiments have been conducted in both constructed complex unbalanced dataset and in the real-world Internet environment. All the results show that our proposed method can avoid problems caused by the spam techniques and is efficient enough to detect large amount of malicious websites, even in the real Internet environment.

III. PROPOSED METHOD

As mentioned earlier, we select end users' perspective to obtain the needed information and use a CNN to analyze captured webpage images for malicious websites detection. The proposed method is further introduced below.

A. WEBPAGE SCREENSHOT

Nowadays, a construction of web pages becomes more and more intricate and the content becomes more and more abundant as well, which contains a large amount of information to identify. Beyond that, as has been mentioned, criminals design websites with various spam techniques such as redirection, embedding, and hiding in order to escape detection. However, in spite of the advanced spam techniques, what does matter to judge a malicious website lies in the webpage finally presented to end users by browsers. It means if the webpage shows malicious content, it is a malicious website. Thus, all the complexities caused by diverse website content and construction techniques can be simplified and the way to detect malicious websites is pointed out that the webpage eventually rendered by the browser is what is really needed by a detection model.

When crawling a webpage, we can only obtain the website's source code. In other words, we cannot get dynamically loaded information generated by JavaScript just from the source code. Luckily, there is an interface called WebDriver which enables developers to create automated tests imitating real user interactions [27]. It offers a language independent protocol that provides remotely controlling the conduction of web browsers [28]. WebDriver offers navigations to web pages, user input, JavaScript execution, and other support. Servers often used to implement WebDriver's wire protocol such as ChromeDriver, FireFoxDriver, and PhantomJS. For the purpose of a fast screenshot speed, we select a headless Web driver scriptable with JavaScript to realize multi-threaded concurrency. Fig. 2 presents an example of a screenshot of a phishing website about a giving away of Bitcoins.

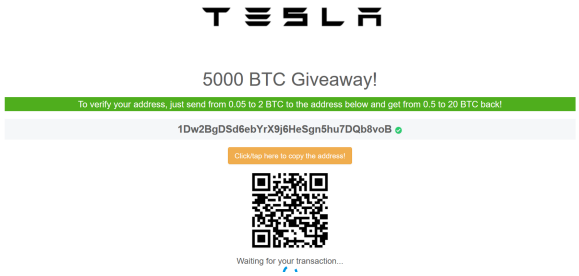

FIGURE 2. Sample malicious website screenshot.

B. DETECTION MODEL

A CNN, just as its name has implied, means that it is a network that uses convolution; to be more specific, it uses convolution instead of matrix multiplication in at least one layer [29]. LeCun *et al.* in 1998 first proved high performance of a CNN model for handwriting recognition [30]. Since then, CNNs have been becoming more and more popular in more different fields, such as natural language processing [31], disaster climate discovery [32], and clinical medicine [33], which all proved the great performance of the CNNs. What is more, to be noted, a CNN is used to analyze visual images at the very beginning and it can reduce the complexity of the model by sharing convolution weight and in a way help solving an overfitting problem.

There are two main reasons why we adopt a CNN to analyze and classify the captured images of malicious website:

1) Manual feature extraction models are often used for image category. However, nowadays the CNN proves its outstanding performance in such tasks, especially when compared with the traditional models in categorizing large numbers of samples.
2) Despite a long time of the CNN training process, it has a fast detection speed, which can solve the detection problem caused by the rapid change of numerous malicious websites.

In order to make the detection more efficient on the constructed unbalanced and complex dataset and in the real Internet environment, after studying on the structures of the classical deep learning models such as AlexNet, VGGNet, GoogLeNet, and Deep residual learning [34]–[37], we adopted the following CNN model structure shown in Fig. 3.

The model structure is as follows:

- The input layer is a screenshot of the page with the size adjusted. The size is $W * L * 3$, where 3 represents the three channels.
- The first convolution layer contains M_1 convolution kernels, whose size is $K_1 * K_1$, and it is activated by a Rectified Linear Unit (ReLU) and maximally pooled. The pooling window is (P_1, P_1).
- The second convolution layer contains M_2 convolution kernels, whose size is $K_2 * K_2$, and it is activated by a

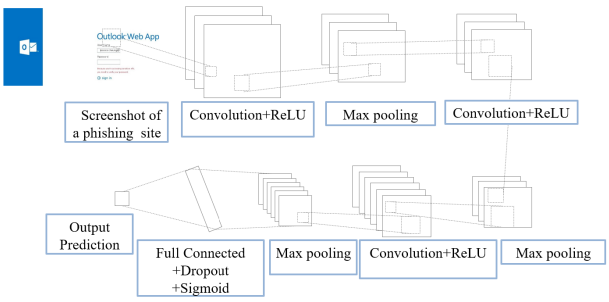

FIGURE 3. Used CNN model structure.

ReLU and maximally pooled. The pooling window is (P_2, P_2).

- The third convolution layer contains M_3 convolution kernels, whose size is $K_3 * K_3$, and it is activated by a ReLU and maximally pooled. The pooling window is (P_3, P_3).
- The full connection layer has N neurons, and is activated by ReLU. It prevents overfitting through dropout and further output labels through a sigmoid function.

This model is composed of three spatial convolution layers and one fully connected layer. It creates batches of augmented data so that training data can be enriched and overfitting can be prevented. In addition, we use the random rotation, flip, shift, shear and so on as data augmentation approaches. The size of input images is adjusted to $W * L * 3$, where $W = L = 256$.

Each of the first two convolutions has 32 output filters (i.e., $M_1 = M_2 = 32$), and the third one has 64 output filters (i.e., $M_3 = 64$). The size of the convolution kernel is $3 * 3$, where $K_1 = K_2 = K_3 = 3$. The output of the convolution layer is then mapped nonlinearly. We choose a ReLU shown in Eq. (1) as an activation function for the CNN as it provides fast convergence speed and can easily find the gradient.

$$ReLU(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0. \end{cases} \tag{1}$$

We set the max pooling layers of the model as 3 and the pool size as (2, 2), where $P_1 = P_2 = P_3 = 2$, which makes the input in half on both spatial dimensions.

In addition, a full connection layer with 64 neurons (i.e., $N = 64$) and is connected after several convolution and pooling layers. The convolutional layer and pooling layer are regarded as the process of an automatic image feature extraction. After the image features are extracted, the fully connected layer is used to realize a classification task. Then, at the fully connected layer, the neural network randomly discards half units during the training phase.

The prediction output layer accounts for image prediction and determines the corresponding website category. Considering that it is a binary classification problem, we adopt the sigmoid function shown in Eq. (2).

$$sigmoid(x) = \frac{1}{1 + e^x}. \tag{2}$$

If the sigmoid value is bigger than 0.5, the corresponding site is classified as a phishing website; otherwise, it is recognized as a legitimate website.

In order to match the sigmoid activation function, the following loss function is used in our CNN model.

$$loss(x, z) = -\sum_i (x[i] * log(z[i]) + (1 - x[i]) * log(1 - z[i])), \tag{3}$$

where x is the output and z is the target. The advantage of binary cross-entropy as a loss function is that the sigmoid function can avoid the problem of a reduced learning rate of the mean square error loss function when the gradient descends, because the learning rate can be controlled by the error of the output.

The optimizer used in the CNN model is an Adam optimization algorithm, which is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing.

IV. EVALUATION

In order to verify the effectiveness of the method, we conducted two kinds of experiments: one is conducted on a constructed complex dataset and the other is conducted in the real-world Web environment.

A. EXPERIMENT 1: EVALUATION BASED ON THE CONSTRUCTED DATASET

To evaluate the validity of the proposed method, we first constructed a complex dataset and evaluated the proposed method on it. We then compared the experiment result of our method with other detection methods on the same dataset.

1) CONSTRUCTED COMPLEX DATASET

To test whether the proposed method is effective and practical, a complex data set is constructed in this paper, and its complexity is demonstrated as follows:

- Malicious website data covers socially reported data, data provided by search engines, randomly and manually selected data by the host and so on, which, to be more specific, includes gambling, pornographic, fishing samples, and the like. What is more, about 36% of the collected malicious websites use the three types of spam techniques discussed in the first section, which for most classical machine learning methods, it is very difficult to detect, for as mentioned before, the parsed web content is fine for parser but not the case at all for end users.
- Legitimate website data consists of DMOZ data [38], search engine data and data manually labeled through access to random hosts. The complexity lies in the fact that many positive samples also use text/hyperlink hiding techniques and they sometimes contain sensitive words as well, so this usually misleads detection and leads to a high false positive rate.

- In addition to the multiple spam techniques mentioned above, there are still some positive websites that are difficult for even manual auditors to identify. To collect our data, we invited three volunteers to label each website, and whether it is a malicious website follows the rule that the minority is subordinate to the majority. It is not surprising to find that the decisions made by three volunteers are sometimes inconsistent, for some websites are collected by search engines because of sensitive words they contain. Besides, even after training, the volunteers' perceptions of malicious sites still vary from one to another.

On one hand, these kinds of data greatly increased the difficulty of detection; however, on the other hand, because they are very confusing for most current detection models and even human beings, it helps make the data more like those in the real Internet world nowadays.

The constructed dataset includes 6,104 samples in total, with 2,375 samples of malicious websites and 3,729 samples of legitimate websites. This complex and unbalanced dataset makes sure the evaluation of the proposed method and the further comparative experiment between this method and traditional machine learning methods valid and reliable. In addition, it makes the proposed method suitable for the detection in the real Internet environment as well.

2) DETECTION METRICS

The detection metrics used in this paper include the accuracy, precision, True Positive Rate (*TPR*), False Positive Rate (*FPR*), F1-Measure (F_1), loss, and Area Under Curve (*AUC*). Before discussing these detection metrics, we first clarify the concept of the basic terms as: 1) true positive refers to correctly predicted event values; 2) false positive refers to incorrectly predicted event values; 3) true negative refers to correctly predicted no-event values; and 4) false negative refers to incorrectly predicted no-event values. Then, a confusion matrix is made for summary in Table 1.

With the clear definitions of these terms, we can further make clear of the detection metrics:

- Accuracy calculated as $\frac{TP+TN}{TP+TN+FP+FN}$ is the most intuitive measure of performance, which is simply the ratio of correctly predicted items to total items.
- Precision calculated as $\frac{TP}{TP+FP}$ is the ratio of correctly predicted positive items to the total predicted positive items.
- *TPR* (also known as recall) calculated as $\frac{TP}{TP+FN}$ is the ratio of correctly predicted positive items to all the positive items that should be predicted.
- *FPR* calculated as $\frac{FP}{FP+TN}$ is the ratio of falsely predicted positive items to all the negative items that should be predicted.
- F_1 calculated as $\frac{2*(Recall*Precision)}{Recall+Precision}$ is the weighted average of the precision and the recall. Therefore, this metric takes both false positive and false negative into account. Although it is not as easy to understand as accuracy, the F1-measure is usually more useful than the accuracy, especially when you have an uneven class distribution.
- Loss is calculated on training and validation when training a CNN model. The loss value indicates how well or poorly a certain model behaves after each iteration of optimization. The lower the loss, the better a model is, under the condition that the model has not over-fitted to the training data. Ideally, one would expect the reduction of loss after several iterations. Unlike the accuracy, the loss is not a percentage. It is a summation of the errors made for each example in training or validation sets.
- *AUC* provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting *AUC* is as the probability that the model ranks a random positive example more highly than a random negative example.

TABLE 1. Confusion matrix.

		Reality (Truth)	
		event	no-event
Predicted Label	event	True Positive (TP)	False Positive (FP)
	no-event	False Negative (FN)	True Negative (TN)

3) EVALUATION OF THE PROPOSED METHOD BASED ON THE DATASET

Keras is used to setup the experiments [39]. Keras is an advanced neural network API written in Python which runs on TensorFlow [40]. The CNN structure used in the experiments are shown in Fig. 3 in Section III and of course all the parameters, such as the convolution kernel size, pool size and so on are also set the same as the values discussed in the section. In addition, *batch_size* is set as 32, and epochs are set as 61 in our experiment. The parameters for image data augmentation are also set: $rescale = 1/255$, $shear\_range = 0.2$, $zoom\_range = 0.2$, $horizontal\_flip = True$.

We randomly select 60% of the samples as training set, 20% of the samples as validation set, and 20% of the samples as test set. Then the training set has 3,662 samples, the validation set has 1,221 samples, and the test set has 1221 samples. The above random sampling process was carried out three times, and three groups of training, verification and test data were obtained. In order to reflect the performance of the algorithm in a more objective way, the data shown in this section are the average values of the experimental results obtained on three sets of data. Fig. 4 shows the loss and accuracy rate of the proposed method based on the training set and validation sets.

The horizontal axis in Fig. 4 represents the number of times epoch. One epoch is one forward pass and one backward pass of all the examples. Fig. 4 shows the change of loss, accuracy, F_1 and *AUC* as the number of epochs increases. On the training set, with the number of iterations increasing, the *loss* value decreases and the *accuracy* increases continuously. When the number of iterations is less than 11, the loss value on the validation set is similar to that on the training set and so

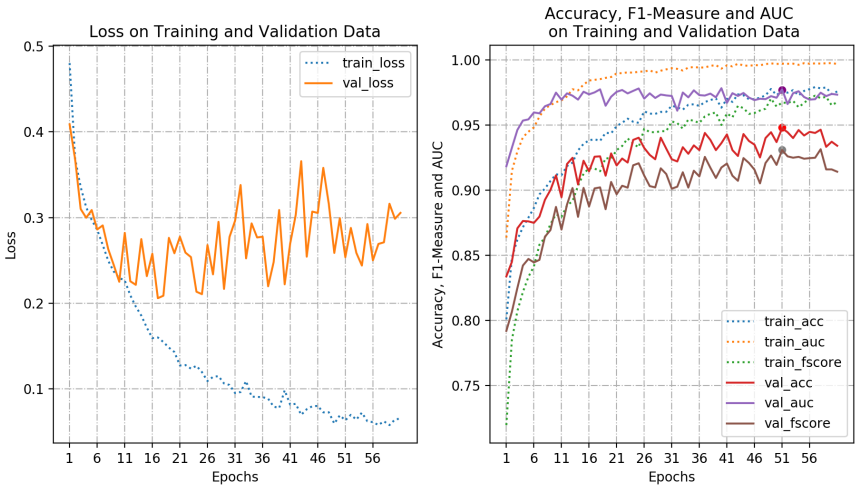

FIGURE 4. Performance of the proposed method based on the training and validation sets.

TABLE 2. Comparisons of different malicious website detection algorithms.

Description	Method	$Recall/TPR$	FPR	$Precision$	F_1
HTML Text Categorization	Naive Bayes	0.7502	0.1747	0.7333	0.7411
	Random Forest	0.5579	0.0801	0.8165	0.6626
	Logistic Regression	0.7452	0.1180	0.802	0.7721
	SVM	0.7417	0.1033	0.8214	0.7792
Screenshot Recognition	SIFT+SVM	0.7312	0.4060	0.5375	0.6181
	CNN	**0.9361**	**0.0531**	**0.9184**	**0.9271**

are the accuracy, F_1 and *AUC* values; but when the number of iterations is more than 11 times, the *loss* value begins to oscillate slightly on the validation set, and *accuracy*, F_1 and *AUC* values shows a slow growth trend on the validation set until the iterations reaches 51 times. With more iterations, *accuracy*, F_1 and *AUC* on the validation set begin to decline. That is to say, the model is over fitting after 51 iterations. Therefore, in the following comparative experiments on the testing set, the iteration is set as 51 times.

4) COMPARISON OF THE PROPOSED METHOD WITH OTHERS

Comparisons between our method and other methods on the testing set were made to further test the effectiveness and practicability of the proposed method. Other methods refer to the classical feature-based machine learning methods, that is, using traditional machine learning methods to analyze website information obtained through page crawling, similar to the way that commercial search engines collect data. In this experiment, we chose some commonly used traditional machine learning methods, including Naive Bayes, Random Forest, Logistic Regression, SVM, and Scale-Invariant Feature Transform (SIFT)+SVM. Table 2 presents the comparative result of different malicious website detection methods. The models are all trained on the training set, and the experiment results shown in the following table are the results on the testing set.

As shown in Table 2, among the traditional methods, SVM shows good performances with "0.8214" in precision and "0.7792" in F_1, but still much lower than CNN that provides "0.9184" for precision and "0.9271" for F_1. Naive Bayes has the high recall rate of "0.7502", but still much lower than CNN. Random Forest performs good in FPR with "0.0801", but still higher than CNN, which provides "0.0531" for FPR. The result that traditional methods using text classification by page crawling is not satisfying on testing set is mainly caused by the popularity of spam techniques. However, the script-enable text recognition method works much better. It is because, through JavaScript parsing, the first two types of spams above can be effectively solved, but it is still ineffective for content and hyperlink hiding techniques.

Compared with the above methods, the detection method based on visual screen images can effectively solve various types of cloaking and hidden spams. Particularly, due to the good performance of the CNN, the proposed screenshot image recognition method surpasses the classical feature-based machine learning methods and has shown the best performance. All the experiment results prove that the proposed method can work effectively in malicious website detection.

B. EXPERIMENT 2: EVALUATION OF THE PROPOSED METHOD IN THE REAL WEB ENVIRONMENT

In order to verify the malicious website discovery ability of the proposed algorithm in the complex real-world Web environment, a system based on the proposed method was developed, and the system was put into practice for three months.

Although the proposed method shows good performance on the collected dataset, it will take a very long time for large-scale webpage screenshots in the real-world Web environment. Because although headless Chromedriver can be used in multi-threaded situations, the screenshot ability of stand-alone server is still limited. Consequently, if having a cluster of dozens of servers, the method proposed in this paper can be directly used as an independent and complete detection

solution. However, if the number of servers is limited and the amount of data to be detected is particularly large (e.g., greater than 10 million websites), it is better to use some other filtering methods first, and use the proposed method at later stage. Therefore, we used the domain name registrant and domain name resolution server etc. to lock the suspected malicious website targets in the previous stage. In other words, simple content independent analysis methods were used to quickly filter out the suspected malicious websites, and the detection was then conducted using the proposed method.

We put the proposed method into practice and the result was encouraging. A malicious website discovery system was developed based on the proposed method and was deployed on five machines. The detection data source contained nearly 20 million websites. From March 1st, 2018 to May 31th, 2018, the system discovered 96,236 malicious websites. Then the discovered websites were reported to National Domain Name Complaint Processing Center (NDNCPC). According to NDNCPC's feedback, 91,357 reports were confirmed as malicious websites. During this period, a total of 97,635 reports were identified as malicious websites by NDNCPC. In other words, 93.55% of malicious websites reports were from our system, which means it has become the main discovery channel of malicious websites of NDNCPC. It should be noted that usually the service of malicious websites is not stable, so for example, those websites that were not identified as malicious are mostly because they were not accessible when they were manually identified.

V. DISCUSSION

The experiments in Section IV show that the proposed method can effectively combat malicious websites using Web spam techniques. The reason is that the proposed method selects end users' perspective, which invalidates Web spam behaviors at all hidden middle layers. But it is worth mentioning that, the proposed approach does not distinguish which of the identified malicious websites employ spam techniques. Then, one question arises: Can we first determine which sites use spam techniques, and then design corresponding algorithms to detect whether it is a malicious website? This is a seemingly reasonable logic, but it is quite difficult to really make it happen. On the one hand, all kinds of spams, especially the spams using JavaScript techniques, are extremely fraudulent. At present, there is no effective detection algorithm for all kinds of spams. On the other hand, even if a website is clearly confirmed using spam techniques, it is still difficult to further determine whether it is a malicious application or not.

What is more, since the proposed method uses the CNN and aims at large scale data, the speed or the cost is worthy of discussion. The running time of a CNN includes two parts, training and testing. The training time is relatively long but is acceptable especially with the use of powerful graphics processing units. Once the training is done, the testing is efficient so generally the time cost is not a problem for this method.

In addition, another issue is also worth discussing. Since the proposed approach can invalidate various spam techniques in detecting malicious sites such as pornography, gambling and phishing, can it be extended to other types of website recognition, such as website classification? In fact, the answer to this question is an obvious yes. As long as the website detection or classification problem has visual similarity on the user-side presentation, the proposed method can be used for reference.

At last, besides the application in the second experiment, is there any applicable scenario for the proposed method? The answer is yes. Because thanks to the launch of the Google TensorFlow.js project in 2018, which is a JavaScript library for training and deploying machine learning models in the browser, browser plug-ins will become a more reasonable application mode. When users access web pages through browsers, all needed page resources are loaded. Besides webpage screenshots, the proposed method does not consume extra resources. Therefore, as a browser extension, it will filter the phishing websites conveniently.

VI. CONCLUSION AND FUTURE WORK

In this paper, we discussed several typical Web spam techniques that often confuse malicious website detection and lead to inefficiency. Then to solve the problems caused by spam techniques and also to make the detection more efficient, we adopted the end-user's perspective and use the CNN to learn and recognize screenshot images of the websites. This novel method proved to be very efficient by both a comparative experiment on the constructed complex dataset and the experiment set in the real Internet environment.

Our future research may include: (1) The screenshot ability can be further enhanced so that the method can be even faster and more adaptable to large-scale detection, especially to the real Web environment; (2) This research only focuses on the image feature and future research may merge both image and text features to build a more robust detection model; (3) Although this research uses the proposed method to detect malicious website, this method is in fact a classification technique, so future research may apply this method to other different types of website classification tasks; and (4) At last, to make the method more practical and user-friendly, the algorithm of the proposed method may be further developed as a browser plug-in and carry out real tests as a browser extension.

REFERENCES

[1] A. Ranjan. (2019). *The Battle Against Malicious Websites*. [Online]. Available: https://www.techtricksworld.com/the-battle-against-malicious-websites/

[2] S. C. Dombrowski, K. L. Gischlar, and T. Durst, "Safeguarding young people from cyber pornography and cyber sexual predation: A major dilemma of the Internet," *Child Abuse Rev., J. Brit. Assoc. Study Prevention Child Abuse Neglect*, vol. 16, no. 3, pp. 153–170, May 2007.

[3] K. Chellapilla and A. Maykov, "A taxonomy of JavaScript redirection spam," in *Proc. 3rd Int. Workshop Adversarial Inf. Retr. Web (AIRWeb)*, 2007, pp. 81–88.
[4] G.-G. Geng, X.-T. Yang, W. Wang, and C.-J. Meng, "A taxonomy of hyperlink hiding techniques," in *Proc. Asia–Pacific Web Conf.* Cham, Switzerland: Springer, 2014, pp. 165–176.
[5] D. Liu, J.-H. Lee, W. Wang, and Y. Wang, "Malicious websites detection via CNN based screenshot recognition," in *Proc. Int. Conf. Intell. Comput. Emerg. Appl. (ICEA)*, Aug. 2019, pp. 115–119.
[6] Y. Fukushima, Y. Hori, and K. Sakurai, "Proactive blacklisting for malicious Web sites by reputation evaluation based on domain and IP address registration," in *Proc. IEEE 10th Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Nov. 2011, pp. 352–361.
[7] A. Dewald, T. Holz, and F. C. Freiling, "ADSandbox: Sandboxing JavaScript to fight malicious websites," in *Proc. ACM Symp. Appl. Comput. (SAC)*, 2010, pp. 1859–1864.
[8] J. Zhang, P. Porras, and J. Ullrich, "Highly predictive blacklisting," in *Proc. USENIX Secur. Symp.*, 2008, pp. 107–122.
[9] M. S. Hanif and M. Bilal, "Competitive residual neural network for image classification," *ICT Exp.*, vol. 6, no. 1, pp. 28–37, Mar. 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405959519300694
[10] G. Swapna, R. Vinayakumar, and K. P. Soman, "Diabetes detection using deep learning algorithms," *ICT Exp.*, vol. 4, no. 4, pp. 243–246, Dec. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405959518304624
[11] R. Vinayakumar and K. P. Soman, "DeepMalNet: Evaluating shallow and deep networks for static PE malware detection," *ICT Express*, vol. 4, no. 4, pp. 255–258, Dec. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405959518304636
[12] B. Altay, T. Dokeroglu, and A. Cosar, "Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection," *Soft Comput.*, vol. 23, no. 12, pp. 4177–4191, Jun. 2019.
[13] G.-G. Geng, X.-D. Lee, and Y.-M. Zhang, "Combating phishing attacks via brand identity and authorization features," *Secur. Commun. Netw.*, vol. 8, no. 6, pp. 888–898, Apr. 2015.
[14] S. Marchal, K. Saari, N. Singh, and N. Asokan, "Know your phish: Novel techniques for detecting phishing sites and their targets," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2016, pp. 323–333.
[15] A. Pandey, V. Sharma, S. Paanchbhai, N. Hedaoo, and S. Zade, "Optical character recognition (OCR)," *Int. J. Eng. Manage. Res.*, vol. 7, no. 2, pp. 159–161, 2017.
[16] A. Desai, J. Jatakia, R. Naik, and N. Raul, "Malicious Web content detection using machine leaning," in *Proc. 2nd IEEE Int. Conf. Recent Trends Electron., Inf. Commun. Technol. (RTEICT)*, May 2017, pp. 1432–1436.
[17] S. Gupta and S. Sachdeva, "Invitation or bait? Detecting malicious URLs in Facebook events," in *Proc. 11th Int. Conf. Contemp. Comput. (IC)*, Aug. 2018, pp. 1–6.
[18] A. K. Jain and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," *Telecommun. Syst.*, vol. 68, no. 4, pp. 687–700, Aug. 2018.
[19] A. M. P. Braşoveanu and R. Andonie, *Semantic Fake News Detection: A Machine Learning Perspective*. Cham, Switzerland: Springer, 2019.
[20] K. Shu, D. Mahudeswaran, and H. Liu, "FakeNewsTracker: A tool for fake news collection, detection, and visualization," *Comput. Math. Org. Theory*, vol. 25, no. 1, pp. 60–71, 2018.
[21] A. Sirageldin, B. B. Baharudin, and L. T. Jung, "Malicious Web page detection: A machine learning approach," in *Advances in Computer Science and Its Applications*. Cham, Switzerland: Springer, 2014, pp. 217–224.
[22] H.-H. Wang, L. Yu, S.-W. Tian, Y.-F. Peng, and X.-J. Pei, "Bidirectional LSTM malicious webpages detection algorithm based on convolutional neural network and independent recurrent neural network," *Int. J. Speech Technol.*, vol. 49, no. 8, pp. 3016–3026, Aug. 2019.
[23] N. B. Harikrishnan, R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Time split based pre-processing with a data-driven approach for malicious URL detection," in *Cybersecurity and Secure Information Systems*. Cham, Switzerland: Springer, 2019, pp. 43–65.
[24] S. Smadi, N. Aslam, and L. Zhang, "Detection of online phishing email using dynamic evolving neural network based on reinforcement learning," *Decis. Support Syst.*, vol. 107, pp. 88–102, Mar. 2018.
[25] Y. Peng, S. Tian, L. Yu, Y. Lv, and R. Wang, "A joint approach to detect malicious URL based on attention mechanism," *Int. J. Comput. Intell. Appl.*, vol. 18, no. 03, Sep. 2019, Art. no. 1950021.
[26] Z. Fa, G.-G. Geng, Z.-W. Yan, and X.-D. Lee, "A robust Internet abuse detection method," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 1712–1715.
[27] S. Avasarala, *Selenium WebDriver Practical Guide*. Birmingham, U.K: Packt, 2014.
[28] *Webdriver w3c Living Document*. Accessed: Feb. 29, 2020. [Online]. Available: https://w3c.github.io/webdriver/
[29] I. Goodfellow, Y. Bengio, A. Courville, and F. Bach, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[30] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[31] J. Cui, J. Long, E. Min, Q. Liu, and Q. Li, "Comparative study of CNN and RNN for deep learning based intrusion detection system," in *Proc. Int. Conf. Cloud Comput. Secur.* Cham, Switzerland: Springer, 2018, pp. 159–170.
[32] S. N. K. B. Amit, S. Shiraishi, T. Inoshita, and Y. Aoki, "Analysis of satellite images for disaster detection," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2016, pp. 5189–5192.
[33] H. Jin, Z. Li, R. Tong, and L. Lin, "A deep 3D residual CNN for false-positive reduction in pulmonary nodule detection," *Med. Phys.*, vol. 45, no. 5, pp. 2097–2107, May 2018.
[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[35] P. M. Cheng and H. S. Malhi, "Transfer learning with convolutional neural networks for classification of abdominal ultrasound images," *J. Digit. Imag.*, vol. 30, no. 2, pp. 234–243, Apr. 2017.
[36] P. Tang, H. Wang, and S. Kwong, "G-MS2F: GoogLeNet based multi-stage feature fusion of deep CNN for scene recognition," *Neurocomputing*, vol. 225, pp. 188–197, Feb. 2017.
[37] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[38] DMOZ. *Open Directory Project*. Accessed: Feb. 29, 2020. [Online]. Available: https://w3c.github.io/webdriver/
[39] F. Chollet, "Keras: Deep learning library for theano and tensorflow.(2015)," Github, San Francisco, CA, USA, Tech. Rep., 2015. [Online]. Available: https://github.com/nitbix/keras-oldfork/blob/master/README.md
[40] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, and X. Zhang, "Tensorflow: A system for large-scale machine learning," USENIX Assoc., Savannah, GA, USA, Tech. Rep., 2016.

DONGJIE LIU is currently pursuing the Ph.D. degree with the Computer Network Information Center, Chinese Academy of Sciences, and also with the University of Chinese Academy of Sciences, Beijing, China. Her research interests include machine learning and web data analysis.

JONG-HYOUK LEE (Senior Member, IEEE) received the Ph.D. degree in computer engineering from Sungkyunkwan University, Suwon, South Korea. He is currently leading the Protocol Engineering Laboratory, Sejong University. He is also an author of the Internet Standards: IETF RFC 8127, IETF RFC 8191, and IETF RFC 8691. His research interests include protocol engineering and security. He received the IEEE Best Land Transportation Paper Award, in 2015, the Haedong Young Scholar Award, in 2017, and the IEEE Systems Journal Best Paper Award, in 2018.