LSTM Based Phishing Detection for Big Email Data

Qi Li, Mingyu Cheng, Junfeng Wang, and Bowen Sun

Abstract—In recent years, cyber criminals have successfully invaded many important information systems by using phishing mail, causing huge losses. The detection of phishing mail from big email data has been paid public attention. However, the camouflage technology of phishing mail is becoming more and more complex, and the existing detection methods are unable to confront with the increasingly complex deception methods and the growing number of emails. In this article, we proposed an LSTM based phishing detection method for big email data. The new method includes two important stages, sample expansion stage and testing stage under sufficient samples. In the sample expansion stage, we combined KNN with K-Means to expand the training data set, so that the size of training samples can meet the needs of in-depth learning. In the testing stage, we first preprocess these samples, including generalization, word segmentation and word vector generation. Then, the preprocessed data is used to train a LSTM model. Finally, on the basis of the trained model, we classify the phishing emails. By experiment, we evaluate the performance of the proposed method, and experimental results show that the accuracy of our phishing detection method can reach 95 percent.

Index Terms—Phishing email, LSTM, social engineering

1 Introduction

In recent years, cyber security incidents have occurred frequently. In most of these incidents, attackers have used phishing email as a knock-on to successfully invade government systems (such as the US State Department and the White House [1]), well-known companies (such as Google and RSA [2]), and websites of politicians and social organizations in many countries (such as John Podesta and DNC [3]). This series of high-profile incidents highlights the growing popularity and power of phishing attacks. On the one hand, phishing emails often cause economic damage to enterprises. On the other hand, phishing emails lead to the leakage of private information, which causes damage to the industry or even the country.

Unlike attacks that exploit specific technical vulnerabilities in software and protocols, phishing attacks are based on social engineering [4], [5]. By sending fraudulent emails, the attacker induces the recipient to take some dangerous actions (such as clicking on links, entering passwords, etc.) without knowing it. From the attacker's point of view, phishing attack does not need too much technical cost, does not depend on any specific vulnerabilities, and is easier to avoid technical defense than malware attack. From the defender's point of view, phishing attack is difficult to be judged by simple rules, and it is difficult to achieve all-round protection [6], [7]. Meanwhile, with the widespread use of mail, the number of mails has increased, and traditional methods are difficult to detect phishing emails on big email data efficiently. For these reasons, there is no universal and effective tool to detect or prevent harpoon phishing, which makes it as the main attack means to break through valuable targets, and has also been proved to be the prelude of most APT attacks [8]. In recent years, phishing attacks have attracted more and more attention. Researchers try to use sandbox [9], behavior blacklist [10], [11], filtering botnet email [12], sender reputation analysis [13], [14], linguistic attribution [15], [16] and other methods to analyze phishing behavior [17]. Among these existing research methods, behavior blacklist method, botnet email filtering method and sender reputation analysis method need to be isolated according to sender's information. However, these analysis methods cannot confront with the impact of increasingly complex phishing attacks in the confrontation environment, for the research of phishing involves many aspects besides cyber-attacks, such as social engineering [18], psychology [19], economics [20], consciousness [21], and coping measures [22]. These technologies are mainly used to prevent fraudulent phishing which redirects users to fake websites through embedded links in e-mail, and is not easy to adapt to activity attribution and identification.

Machine learning [23], [24] is an effective way to solve phishing attacks, when introduced into phishing email detection in complex environments. But this idea faces many difficulties in the actual implementation process. Importantly, in practice, phishing email can be classified into many categories [25] according to its camouflage means, such as disguising as a public domain name, copying IP, using short links, etc., each kind of phishing email has different characteristics.

- *Q. Li is with the Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: liqi2001@bupt.edu.cn.*
- *M. Cheng is with the Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: chengmingyu@bupt.edu.cn.*
- *J. Wang is with the Sichuan University, Chengdu, Sichuan 610065, China. E-mail: wangjf@scu.edu.cn.*
- *B. Sun is with the China Information Security Certification Center, Beijing, China. E-mail: 273908200@qq.com.*

Manuscript received 25 Nov. 2019; revised 25 Feb. 2020; accepted 27 Feb. 2020. Date of publication 12 Mar. 2020; date of current version 14 Jan. 2022.
(Corresponding author: Junfeng Wang.)
Recommended for acceptance by Y. Wen.
Digital Object Identifier no. 10.1109/TBDATA.2020.2978915

Fig. 1. Imitating linked phishing emails and email header.

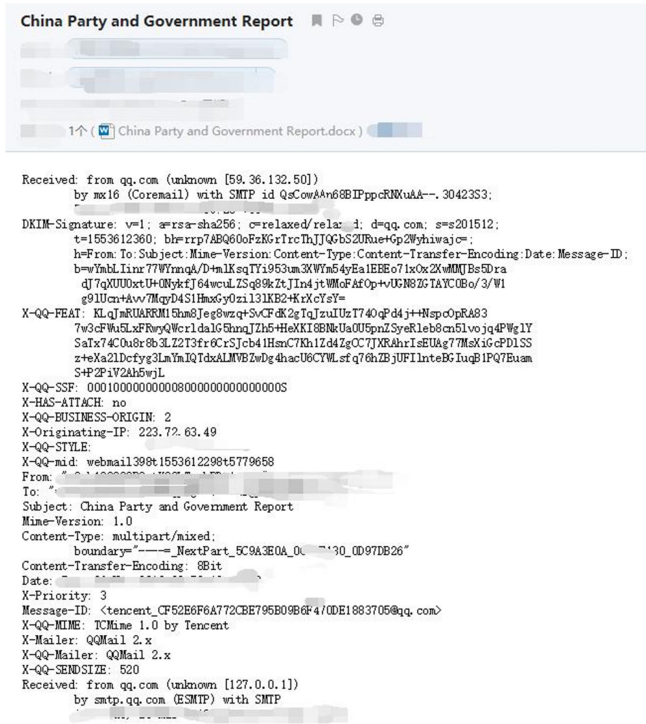

Fig. 2. Malicious attachment phishing email and email header.

Although these methods mentioned above can detect phishing email to a certain extent, for identity forgery and cloud attachment, the methods such as feature extraction and sandbox are invalid. In addition, there is a great difference between the various open-source datasets used for research on the Internet and the real data in practical application, which seriously affects the generalization of the model and detection effect.

Therefore, we first propose a sample labeling method in our paper. We use a clustering algorithm to accurately label the existing email samples on big email data which are not marked precisely. Meanwhile, we can also expand the email samples and solve the problems caused by insufficiently accurately labeled data. Secondly, since we need to classify according to the message body, so we use the LSTM (Long Short-Term Memory Network) neural network model for training, mainly owing to the excessive length of the message body. The LSTM neural network can effectively process information through three gate units, and solve the problem of gradient disappearance caused by the excessive length of context. So, we can train an LSTM neural network model to detect phishing emails, which effectively solve the problems mentioned above and achieve effective detection of phishing emails.

2 Related Work

The essence of phishing email [26] is that by inducing people to click on malicious links or open malicious documents and attachments, the attackers can complete their malicious purpose. Nowadays, the phishing email attack methods are mainly divided into two categories as is shown in Figs. 1 and 2. Fig. 1 shows malicious-link based phishing email and the email's header, which involves constructing a similar domain name or imitating a domain name to attack the recipient, and the recipient will be attacked once the link is clicked [27]. Fig. 2 shows malicious attachment phishing email and email's header, which mainly involves inducing the recipient to download and open the malicious attachment of the email, and using the malicious attachment to attack the victim.

By analyzing phishing emails reported by the sandboxes, these phishing emails have four characteristics, which will help us detecting phishing emails effectively.

1) Flexibility: When an attacker sends a phishing e-mail, email's properties can be changed flexibly. For example, attackers can use various IPs, email's names, domains and malicious files. The domain name may be newly applied, or controlled through a website vulnerability. Moreover, malicious documents used by phishing emails may use system vulnerabilities that have been exposed in the vulnerability library or use unexposed *0day* vulnerabilities, which adds great difficulty to the detection of phishing emails.
2) Broadcastability: Attackers don't care target person, they only care about whether the attack is successful. Therefore, the similar phishing email may be delivered to multiple mailboxes, affecting many people.
3) Inductivity: The contents and themes of phishing emails are diverse, such as news, politics, economy, entertainment gossip, etc. But each category of phishing emails are inductive definitely, that is, phishing emails must induce recipients to click the malicious content in the file, so the phishing emails must be inductive.

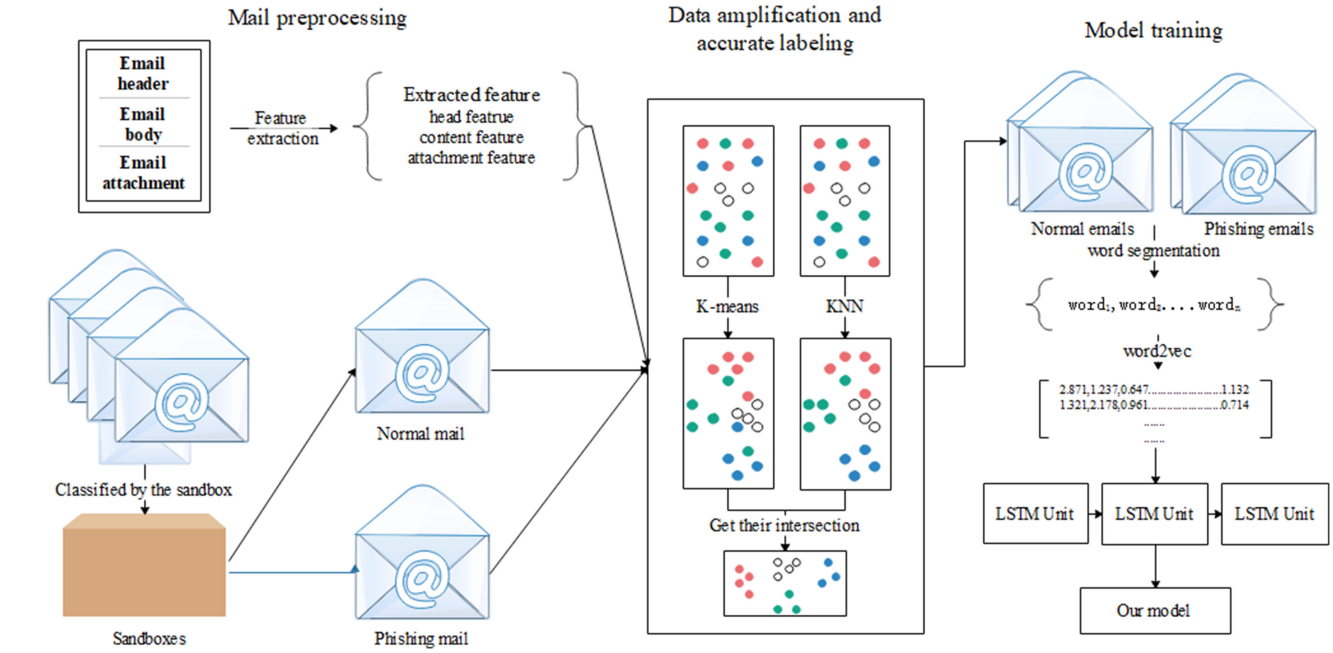

Fig. 3. The framework of the proposed phishing detection method.

4) Severity: Once the link is clicked by the recipient or the attachment is opened by the recipient, the malicious attackers may remotely manipulate the victim's host, steal the confidential file from the compromised host, even spreading the virus through the captured host to the entire intranet environment, which will cause significant harm to the entire enterprise and even national security.

The main methods of detecting phishing emails are as follows: Sandbox-based phishing email detection method [28], [29], [30], black and white list-based phishing email detection method [31], [32], [33], [34], machine learning based phishing email detection method [35], [36], [37], [38].

Sandbox-based phishing email detection method mainly performs static and dynamic detection of phishing email attachments through sandbox to detect phishing emails [29], [30]. In Ref. [30], the author proposed a novel sandbox tool to detect attachments in emails. This tool had a good detection effect on unencrypted malicious documents, but cannot detect encrypted attachments.

Phishing email detection method based on black and white lists [32], [33], [34] detects phishing emails by establishing a feature database for the phishing emails and normal emails [31]. In Ref. [32], a method is proposed for detecting phishing emails based on domain name credibility, by setting the domain name credibility by historical data. However, attackers who use phishing emails can change their IP, sender's mailbox or change the attachment name at their will and get rid of the detection.

The phishing email detection system based on machine learning [35] detects phishing emails by extracting a large number of features and fitting the features using a model of machine learning. In Ref. [39], an anti-phishing method based on feature analysis is proposed to detect phishing emails. The features are extracted by the label of historical data, and information entropy. In Ref. [40], the authors introduce a method with the Natural Language Processing method aiming at detecting phishing emails by extracting keywords from the message body. The main problem of these two methods is that there will be feature loss in the processing of feature extraction. Therefore, machine learning algorithms can not accurately detect phishing emails.

Owing to the difficulty of detection caused by complicated form of phishing email, this paper uses LSTM neural network to automatically extract the characteristics of phishing emails to detect the flexible and versatile phishing emails.

3 Framework Overview

In our proposed method there are two key issues need to be solved. On one hand, the data in our experiments is from enterprise, which has their own mail server. Due to the differences of the type and the distribution between open-source dataset and our data, we cannot use open-source dataset to train machine learning or deep learning model with better generalization. Therefore, we first filter out a certain amount of phishing emails which are in line with the nature of this enterprise manually, and then use KNN and K-Means algorithms to accomplish automatic labelling, which leads the number of samples can support the training of deep learning models. On the other hand, the existing phishing detection methods usually perform feature extraction on the email or separate analysis of the partial sentences of the text information to detect the mail. However, such methods ignore the message body. To this end, after the automatic labelling we use LSTM model of which the input is extended sample to detect the email, which can take advantage of all the information in the e-mail. As shown in Fig. 3, the overall architecture of our proposed system consists of three important phases: Mail preprocessing phase, Data amplification phase and model training phase.

In the automatic labelling phase, we label the emails by the result of the extracted feature, our proposed data labeling method and the pre-classified result. In our method, we proposed a string distance as the distance of our clustering algorithm, and we combine KNN [42] with K-Means [43] to expand the training data set so that the size of training samples can meet the needs of in-depth learning. In the model training phase, we preprocess these samples, including generalization, word segmentation and word vector generation. Finally we train an LSTM model to classify the phishing emails. In our model, Dropout and Regularization is used to avoid over-fitting. We also use Adam as optimizer to adjust learning rate.

4 The Details of Phishing Email Detection

4.1 Email Data Preprocessing Method

Before we label the emails, we proposed a kind of email data preprocessing method to make our methods more efficient. This method is divided into two parts: Filtering offensive emails; labeling daily working mailboxes.

4.1.1 Filtering Offensive Emails

When an attacker intrusion the victim by phishing mails, the attacker's intrusion methods, such as malicious links and malicious attachments, are definitely displayed in the e-mail. Therefore, before classifying the emails, the number of email that needs to be classified can be reduced by filtering the offensive emails, which can greatly improve the efficiency of our algorithm. The way to classify the offensive emails is to make sure that the email contains a URL or attachment. After that, we discard the non- offensive emails and only focus on these offensive emails.

4.1.2 Labeling Daily Working Mailboxes

Some mailboxes send a lot of offensive emails every month. Among these mailboxes, some are normal working mailboxes, others are phishing mailboxes. Therefore, statistics on mailboxes that send a large number of emails per month can effectively reduce the number of emails need to be classified.

4.2 Sample Expansion Method

4.2.1 Automatic Labelling

The data in our paper are collected from an enterprise, which has its own mail server. We found that the mail samples collected from the open-source dataset cannot be applied to the real scenario due to its different distribution. That is to say, we can't use the open-source dataset to train a model and use it to detect the phishing mail in the real world. Therefore, it is very important to label the data of actual data environment and obtain the training dataset that meets the needs. Since data labelling is a relatively complex work, we obtained a small amount of labeled data by artificial method, especially for malicious samples. Then we used KNN and K-Means to expand the samples from the whole dataset on the basis of these small amount of labeled data. Then the expanded samples with high similarity to the manual labeled data will be used for subsequent analysis.

4.2.2 Phishing Email Feature Extraction Algorithm

Phishing emails have the characteristics of broadcastability, so there will be a large number of similar phishing emails in the email server. They maybe come from the same IP or the same Mailbox. And we propose an email feature extraction algorithm based on the seven-tuple set and label emails by this algorithm. The seven-element features are mainly composed of the following two parts:

A) Header feature: This part of the feature includes real source ip, real sender's mailbox, and the consistency of real mailbox and displayed sender's mailbox. Now that the displayed sender's mailbox can be forged, thus the real information of the sender is vital and should be taken into consideration. This part can be obtained from the eml file in the email server.

B) Content feature of the email: This part of the feature includes the title of the email, attachment's name of the email, the attachment suffix of the email, and whether the email contains a URL. If an URL is included, we first determine whether the URL is long or short, and then determine whether the URL is shortened by URL shortener because phishers often use this method in the email.

We can vectorize the email samples based on the seven-tuple phishing email feature extraction algorithm, and then cluster the phishing emails to get accurate labeling training dataset so that our phishing email detection algorithm can identify the phishing emails accurately and efficiently.

4.2.3 Improved Levenshtein Distance

The Edit Distance [41] represents the minimum number of times a single character needs to be deleted, inserted, or replaced from s to t. For two strings a and b, their lengths are $|\,a\,|$ and $|\,b\,|$, their Levenshtein Distance $leva, b(|a|, |b|)$ defined as:

$$leva, b(|a|, |b|) \begin{cases} \max(i, j) & if\ \min(i, j) = 0 \\ \min \begin{cases} leva, b(i-1, j) + 1 \\ leva, b(i, j-1) + 1 \\ leva, b(i-1, j-1) + 1(ai \neq bj) \end{cases} & otherwise \end{cases} \tag{1}$$

when $ai = bj$ is 0, otherwise is 1. $leva, b(i,\ j)$ is the Edit Distance between the first i characters of a and the first j characters of b. The similarity of a, b can be expressed as $Sima, b$:

$$Sima, b = 1 - (leva, b(|a|, |b|)/\max(|a|, |b|)). \tag{2}$$

By this string distance we can cluster emails and get rid of the problem of feature loss, and cluster the emails accurately.

4.2.4 Sample Labeling Algorithm

The K-Means algorithm [43] is the most classic partition-based clustering algorithm. Its central idea is that all the points are clustered centered on k points in space and all clusters update the values of their center iteratively until the best clustering result is obtained. The concrete steps of algorithm are as follows:

Algorithm 1. K-Means Algorithm

Required: K number of the initial point, Ci center Point of i-th cluster, Pi i-th point, Fij i-th feature of Pj, $FCij$ the i-th feature of Cj, CLi the i-th cluster, Ni the number of points in CLi, Na the number of all data, Da, b the distance between point a and point b
Input: $Pi\ (i = 0 \rightarrow Na - 1)$
Output: the category of Pi

Randomly pick k points as the initial point
While Cluster changes or not reach maximum number of iterations.
  For $Pi(i = 0 \rightarrow Na - 1)$ in dataset:
    For $Cj\ (j = 0 \rightarrow k - 1)(j = 0 \rightarrow k - 1)$:

$$DPi, Cj = \sum_{j=0}^{6} SimFij, FCjj \tag{3}$$

    Assign data to the points of cluster with the smallest D
  In $Ci\ (i = 0 \rightarrow k - 1)$
    For $Pm\ (m = 0 \rightarrow Ni)$:
      For $Pn\ (n = m \rightarrow Ni)$:
        $SimFim, Fin$

$$Ci = (2 \times \sum SimFim, Fin/n(n-1))(i = 0 \rightarrow 6) \tag{5}$$

The central idea of the KNN algorithm [42] is when the data and its label in the training set are known, input the test data, compare the features of the test data with the corresponding features to the training set and find the top K data most similar to the training set. The category of the test data is the most frequently occurring classification of k data. The concrete steps of algorithm are as follows:

Algorithm 2. KNN Algorithm

Required: Pi the i-th point, Fij i-th feature of Pj, Na the number of all the points, Ii the nearest K point of Pi, Ni the number of the intial point, Da, b the distance between point a and point b, Pki the Probability of class i
Input: $Pi\ (i = 0 \rightarrow Na - 1)$
Output: the category of Pi

$Pi\ (i = 0 \rightarrow Na - 1)$
  For $Pi(j = 0 \rightarrow Na - 1)$

$$DPi, Ij = \sum_{i=0}^{6} SimFii, Fij \tag{6}$$

  Ascending $DPi, Ij(i = 0 \rightarrow Na - 1, j = 0 \rightarrow Na - 1)$
    Take the first K distance Dk and corresponding i-th point $PRij$ from the ascending results, "j" represents their category

$$Pkc = \sum DPi, PRjc / \sum_{k=1}^{k} DK \tag{7}$$

  Pi belongs to the category with the highest probability value

After all the sample data detected by sandbox, they are divided into phishing emails and normal emails. Due to the high false alarm rate and false negative rate of the sandbox, the data classified by the sandbox as phishing emails contain normal samples, and classified by the sandbox as normal emails contain phishing emails.

Because of the characteristics of the broadcastability, there are a certain number of similar emails in our email server, so they can be clustered effectively by clustering algorithm. The characteristics of these phishing emails are displayed in the form of strings, such as the mail headers and attachment names, so string distance mentioned above is used as the distance of the clustering algorithm. We use K-Means algorithm and KNN algorithm to cluster and reclassify the results of sandbox.

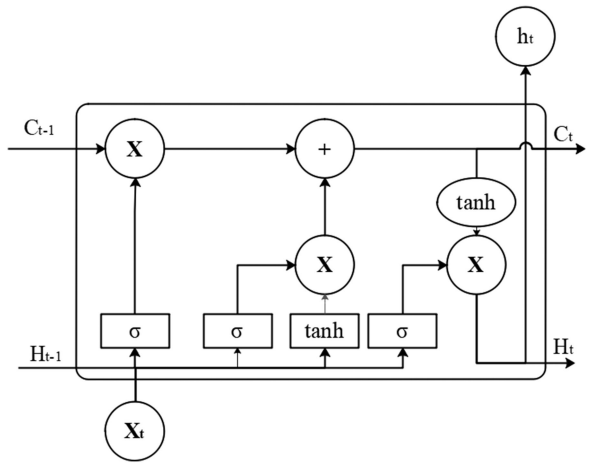

Fig. 4. The structure of LSTM neurons.

The result is defined by the following rules: The first character indicates the type of email judged by sandbox, "p" indicates that the emails are phishing emails, and "n" indicates that the emails are a normal email; The second character indicates the type of email judged by our algorithm, "p" indicates that the emails are classified as phishing emails, and "n" indicates that the emails are judged as normal emails; The third character indicates the algorithm used, "1" is the result of the K-Means algorithm, and "2" is the result of the KNN algorithm. For example, "pp_1" represents emails are phishing emails and they are judged as phishing emails by K-Means algorithm. To ensure the magnitude and reliability of the data set, we use Equations (8) and Equations (9) to get the expanded data set.

Phishing email samples in the dataset:

$$phishing\ samples = pp\_1\ \&\ pp\_2\ +\ np\_1\ \&\ np\_2. \tag{8}$$

Normal email samples in the dataset:

$$normal\ samples = pn\_1\ \&\ pn\_2\ +\ nn\_1\ \&\ nn\_2. \tag{9}$$

In the formula, $a\ \&\ b$ denotes the intersection of a and b, and $a\ +\ b$ denotes the union of a and b.

4.3 LSTM Algorithm

RNN (Recurrent Neural Networks), due to its special network model, considers the previous sequence information when learning the current time information. Therefore, RNN neural network has unique advantages in dealing with time series and text sequence problems. However, it is difficult for RNN to learn long distance information. LSTM is a special form of RNN that overcomes the problems of the classic RNN model [44]. The neuronal cells of the LSTM neural network is shown in Fig. 4.

In Fig. 4, the state of a neuron is similar to a conveyor belt. Data can be transmitted over the entire strip with only

a small amount of linear interaction. It is able to keep the information flowing on it easily. The LSTM neurons are mainly composed of three gate structures that can choose passed information. The gate structure is mainly realized by a neural layer of sigmoid and a point-by-point multiplication operation.

Forgetting Gate: The left part of the Fig. 4 shows the forgetting gate of the LSTM neuron. The input of forgetting gate are $Ht-1$ and Xt. By processing the input, the forgetting gate can output a number between 0 and 1, which represents the degree of forgetting. If the output is 1, it means that all the information is "remembered". If it is 0, it means that all the information is "forgotten".

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f). \tag{10}$$

In Equation (10), Wf is the weights of the forgetting gate, $[ht-1, xt]$ combines two vectors together, bf is the bias of the forgetting gate, σ represents sigmoid function.

Input gate: In the Fig. 4, the middle part represents the input gate, and the input gate layer determines which values need to be updated. The tanh layer generates a new vector, and the input gate layer and the tanh layer update the state together.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i). \tag{11}$$

In (11), Wi is the weights of the input gate, bi is the bias of the input gate

Output Gate: In the Fig. 4, the part on the right is the output gate, which determines the output. Firstly, run a sigmod layer to determine which part of the state is what we need to output. Then, enter the state into tanh function to limit the values of output function between -1 and 1. Finally we can get output by multiplying output obtained in the previous step and sigmoid gate.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o). \tag{12}$$

In (12), $ht-1$ represents the output value of the last neuron. LSTM retains and controls the information completely through these three gate structures. Our paper uses the LSTM algorithm to train the classifier to detect phishing emails.

In our LSTM neuron network, the softsign function is used as the activation function, instead of the tanh function for the faster calculating speed. Adam is used as optimizer to adjust learning rate, so that our model can converge quickly. Orthogonal initialize is also used to solve the gradient disappearance and gradient explosion problem in deep network come from the excessive length of the message body.

After the automatic labelling stage, the number of labeled samples can support the training of deep learning model. Before the message body is used as the input into LSTM, the data is preprocessed as follows:

1. Corpus construction. We construct the corpus of the message body. Ignore the mail without message body.
2. Word segmentation. In this paper, we mainly focus on Chinese mail and English mail. We use Jieba library to implement word segmentation in Chinese sentences and use the space to split English sentences.
3. Removing the stop words. Filter irrelevant words in segmentation results, such as modal particles, auxiliary words and conjunctions.
4. Transformation from words to vectors. In this step, we used word2vec to represent the semantic information of the word by learning the text.
5. Length normalization. First, we calculate the average length of the training data. When the length of a vector is greater than the average length, we truncate the vector. Instead, we fill it with '0'.

5 Experiment Results and Discussion

5.1 Experimental Facilities and Data Sources

In the experiment, we first collect emails from our email server, and mailbox data from some companies and organizations as our experimental data. Email data is selected from January 2017 to June 2018. The total number of the email is 29,942,735. The experiment is conducted in the Ubuntu14.04LTS environment, using python3.5.4 and Keras2.1.2 as the neural network framework to build the network, using Google open source tensorflow1.4.1 as the back-end computing framework. The CPU of the server is Inter(R)Xeon(R)CPU E5-2637v4@3.50 GHz, and the GPU is TITAN(X)(Pascal).

5.2 Evaluation Criteria

In the experiment, the experimental results are evaluated by four parameters, namely Acc, P, R, and F1. These four parameters are defined as follows:

$$Acc = \left(1 - \frac{errorsum}{sum}\right) \times 100\% \tag{13}$$

$$P = \frac{TP}{TP + FP} \times 100\% \tag{14}$$

$$R = \frac{TP}{TP + FN} \times 100\% \tag{15}$$

$$F1 = \frac{2 \times P \times R}{P + R} \times 100\%. \tag{16}$$

In the Equations (13), (14), (15), (16), $errorsum$ represents the number of samples with incorrect classification, sum represents the total number of samples. TP is true positive, it represents the number of phishing emails; FN and FP are the number of false negative and false positive; R represents recall rate, indicating how many phishing emails are correctly classified by the model; F1-score is based on the harmonic mean of the precision and recall rate, evaluating model performance comprehensively.

5.3 Sample Labeling Algorithm Results

In order to verify the results of our sample labeling algorithm, four months are randomly selected from the June 2017 to December 2017 and then 1,000 emails are randomly selected from each month as the verification set to test the labeling method proposed by us.

Since only accurately labeled sample are needed, we verify the accuracy of the results and compare them to the accuracy of the corporate sandboxes. The selected four months are June 2017, September 2017, October 2017, and December 2017. The result is shown in Fig. 5: "pslacc" and "nslacc" means the

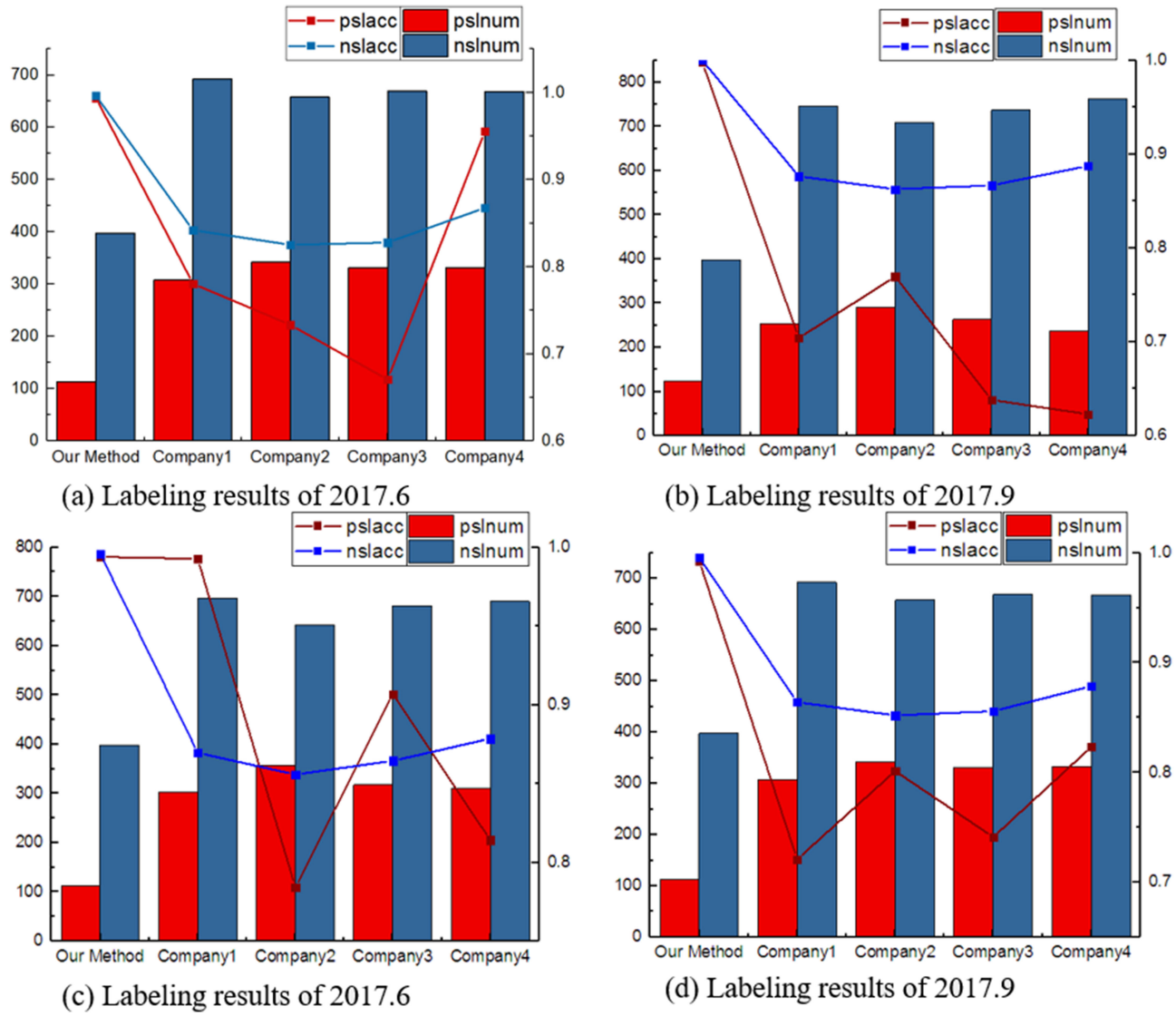

Fig. 5. The results of our labeling method.

accuracy of positive sample and negative sample labeling, "pslnum" and "nslacc" means the number of positive and negative sample labeling. From the Fig. 5, the number of result labeled by our method is slightly less than the number of result labeled by sandboxes, but the accuracy of our labeling algorithm is much higher than the accuracy of result labeled by sandboxes, almost reaches 100 percent.

After analyzing the results of our proposed clustering algorithm, there are always some similarities in the emails that are grouped together, such as similar senders, similar attachment names, or similar mail names. These emails may come from the same attacker, or have similar themes, so these emails are grouped together, which leads to high accuracy of our clustering algorithm. The shortage of our clustering algorithm is settled by our large data volume. The result of our clustering algorithm can effectively support our clustering algorithm and prepare enough accurately labeled email samples for our LSTM neuron network.

Then, the data from June 2017 to December 2017 is used to cluster by our proposed method, and the number of email of type A is 20,3642, the number of email of type B is 10,271, the number of email of type C is 56,920, the number of email of type D is 2,532,984. According to our labeling algorithm:

$$Type\ A = \ pp\_1\&pp\_2 \tag{13}$$

$$Type\ B = pn\_1\&pn\_2 \tag{14}$$

$$Type\ C = np\_1\&np\_2 \tag{15}$$

$$Type\ D = nn\_1\&nn\_2. \tag{16}$$

5.4 LSTM Algorithm Results

In the experiment, the structure of dataset is shown in Table 1, and we first compare the result of other different neurons with the result of LSTM neuron network. These chosen neurons are mainly used to process the sequence data, including standard RNN neurons, GRU neurons, Bi-LSTM neurons and TextCNN neurons. DS_3 as the training set of our model, and 10,000 emails are randomly chosen from January 2018 to June 2018 as the validation set results. The result is shown in Fig. 6. From the Fig. 6, the selected LSTM neurons gets the best result in four neurons. The RNN neural network gets the worst result among the four neural networks due to its simple

TABLE 1
The Number of sAMPLES in Dataset

Dataset	Positive samples		Negative samples		Total samples	Ratio of the positive samples and negative samples
	Type A	Type C	Type B	Type D		
DS1	93080	56920	10271	39729	2000000	3:1
DS2	76414	56920	10271	56395	2000000	2:1
DS3	43080	56920	10271	89729	2000000	1:1
DS4	9746	56920	10271	123063	2000000	1:2
DS5	0	56920	10271	132809	2000000	1:3

model. The results of other four neural network are better than the result of RNN neural networks, but a little poor than the result of our LSTM neural network. Especially, the result of CNN is closest in our method. In the text classification, LSTM and CNN are both commonly used deep learning models. CNN model extracts features similar to n-gram which ignores word order, and it cannot achieve satisfying results in the task of emotion analysis. Relatively, LSTM can better capture the induced statements in the message, thus it works better in phishing detection.

Meanwhile, different datasets are used for our experiment. We use the same LSTM neural network model in the training process. The results are shown in Fig. 7.

From Fig. 7, when ratio of the positive and negative sample is 1:1, the model performed best in the verification set. For imbalanced training data, the prediction accuracy of the model is very unsatisfactory, compared with the balanced training data. The reason is that when the training dataset is imbalanced, the model tilt to the side with the larger sample quantities during the process of training. As we can see in the Fig. 7, the recall rate of DS_1 is very high. The model of DS_1 tends to classify the email as phishing email because of the large number of phishing email in training dataset. But there are a large number of false positives in the results of DS_1, and the F1-score of DS_1 is very low.

At the same time, we input our verification set into four common enterprise sandboxes, and compared the results with the results of our method. From Fig. 8, the enterprise sandboxes perform poorly than our method. The reason is that the enterprise sandboxes cannot detect two kinds of phishing email, malicious link phishing email and malicious attachment phishing email. For the malicious link phishing e-mail, enterprise sandboxes detected such phishing emails mainly by matching links to their information library. However, due to the flexibility of malicious link mentioned above, the enterprise sandboxes cannot effectively detect malicious links, which leads to the low accuracy of sandboxes detection for this type of email. For malicious attachment phishing e-mail, these kind of email is generally used for transfer key

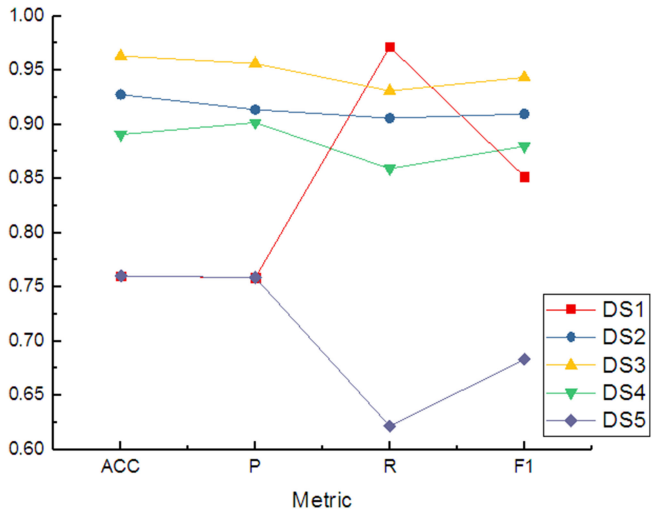

Fig. 7. The results of different datasets.

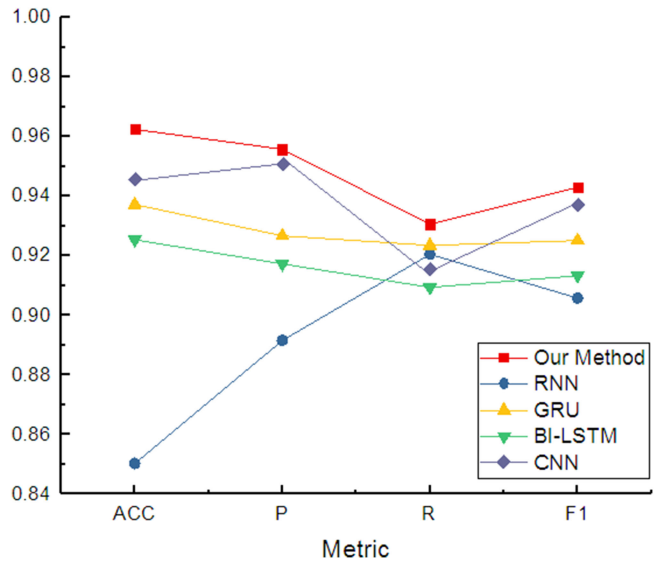

Fig. 6. The results of different neurons.

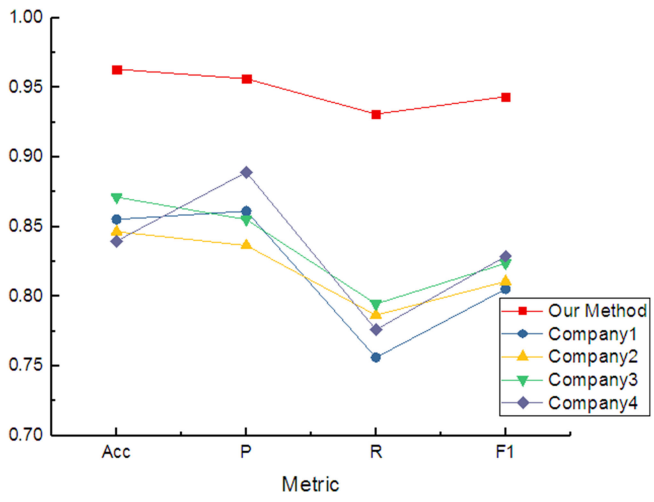

Fig. 8. The results of different datasets.

TABLE 2
Settings of Different Networks

Model \ Hyperparameters	depth	Number of single-layer neurons	Activation function	Regularization	Dropout
M1	5	2048	Relu	L1	0.5
M2	10	1024	Relu	L1	0.5
M3	10	1024	Relu	No Regularization	0.5
M4	10	1024	Relu	L1	0.1
M5	10	1024	leakyrelu	L1	0.5

files, but the enterprise sandboxes cannot effectively detect these two kinds of attachments. These two points are the main reason why the accuracy of the sandboxes is lower than the accuracy of our model.

After that, we adjust the hyperparameters of model, the parameters are set as Table 2.

From the experimental results shown in Fig. 9, the narrow and deep neural network model is more effective when the number of neurons is same. Using L1 regularization can effectively improve the accuracy of the data. In our experimental results, the accuracy of M4 has plummeted, it is because after we reduce the dropout ratio, our training model has a problem of overfitting, so that it performed poorly in the validation set. On the activation function, the accuracy of the model using relu function was slightly higher than accuracy of the model using leakyrelu function.

Then, our method is compared with the machine learning detection model. By extracting the email header features, including the sender IP, the one-hot encoding of the sender country, and whether the sender address and the reply address are consistent, text feature, including one-hot encoding of suffix of the attachment name, we can detect phishing emails by machine learning algorithm and extracted features. We use Support Vector Machines, Random Forest, Xgboost and lightGBM to fit these features and used the model to classify the validation set.

From Fig. 9, the method proposed by us is far superior than the shallow machine learning method. The reasons may be as follow: the features of emails are mainly in the form of strings. When converting the string features into digital features, method is mainly one-hot encoding. However, this method is extremely inefficient in massive data, and a large number of strings cannot be converted into digital features due to feature dimension problems, which leads to loss of a large number of features. Secondly, owing to the flexibility of the phishing e-mail, attackers can change the header features of the emails freely and get rid of phishing email detection system based on machine learning. Ultimately these two reasons lead to the low detection accuracy of shallow learning methods.

Finally, we compare our method to the black and white list method to detect phishing emails. We use the IP, sender, url embedded into email, and attachments from phishing emails and normal emails in the previous months to build black and white list. The result of the black and white list detection is as follows:

As shown in Fig. 10, the detection result used black and white list is far less effective than the result using our method. The phishing email detection method based on black and white list cannot obtain better result, because phishing emails are more flexible. When an attacker replaces the sending mailbox or replaces the attachment name, the black and white list cannot detect that phishing mail.

6 Conclusion

This paper analyzes the existing phishing email detection methods and finds that the traditional detection methods are difficult to accurately detect phishing emails. Therefore, we designed a phishing email detection method based on LSTM neural network. At the same time, when we designed the model, the problem of the phishing email did not have an

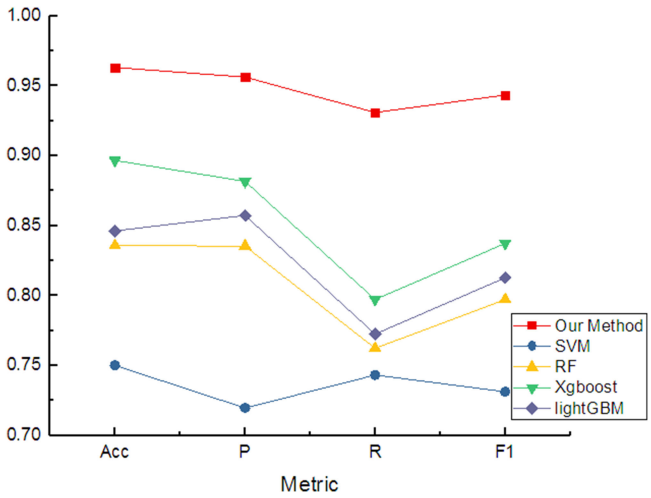

Fig. 9. The results compared with shallow machine learning.

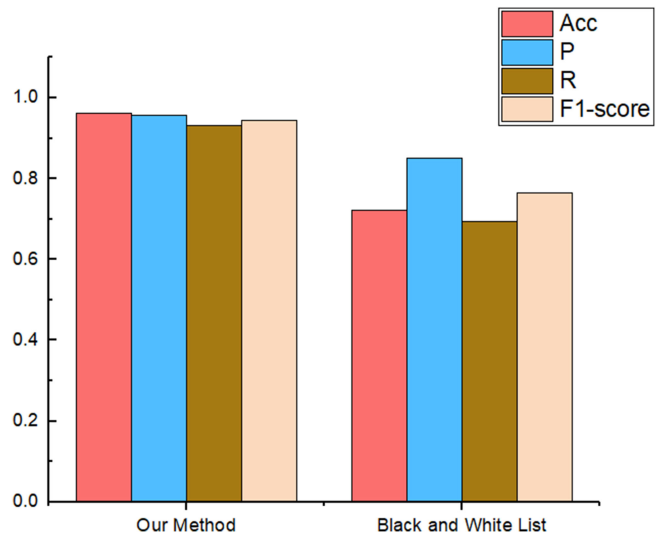

Fig. 10. The results compared with the method of black and white list.

accurately labeled dataset. So we used a phishing email feature extraction algorithm to extract the characteristics of the e-mail, and then use the extracted features to cluster the emails, so as to achieve accurate labeling of phishing emails. Finally, we train the model and compare the proposed method with the traditional phishing email detection method by the experiment. Our method performed better than the existing phishing email detection method, it improves accuracy, reduces false negative rate and false positive rate.

Acknowledgments

This work was supported by the National Key Research and Development Program (2019QY1400, 2018YFB0804503), the National Key Project (GJXM92579), the National Natural Science Foundation of China (U1836103), and the Technology Research and Development Program of Sichuan, China (2019YFG0390).

References

[1] "US State department hack has major security implications," Security Intelligence, 2019. [Online]. Available: https://securityintelligence.com/us-state-department-hack-has-major-security-implications/

[2] K. Zetter, L. Matsakis, I. Lapowsky, G. Graff, E. Dreyfuss, and L. Newman, "Researchers uncover RSA phishing attack, hiding in plain sight," *WIRED*, 2018. [Online]. Available: https://www.wired.com/2011/08/how-rsa-got-hacked

[3] L. Matsakis, I. Lapowsky, G. Graff, E. Dreyfuss, and L. Newman, "Why the DNC thought a phishing test was a real attack," *WIRED*, 2018. [Online]. Available: https://www.wired.com/story/dnc-phishing-test-votebuilder

[4] M. Alsharnouby, F. Alaca, and S. Chiasson, "Why phishing still works: User strategies for combating phishing attacks," *Int. J. Hum.-Comput. Stud.*, vol. 82, pp. 69–82, 2015. [Online]. Available: 10.1016/j.ijhcs.2015.05.005.

[5] T. Jagatic, N. Johnson, M. Jakobsson, and F. Menczer, "Social phishing," *Commun. ACM*, vol. 50, no. 10, pp. 94–100, 2007. [Online]. Available: 10.1145/1290958.1290968.

[6] N. Arachchilage, S. Love, and K. Beznosov, "Phishing threat avoidance behaviour: An empirical investigation," *Comput. Hum. Behav.*, vol. 60, pp. 185–197, 2016.

[7] K. Parsons, A. McCormac, M. Pattinson, M. Butavicius, and C. Jerram, "The design of phishing studies: Challenges for researchers," *Comput. Secur.*, vol. 52, pp. 194–206, 2015.

[8] "Phishing APTs (Advanced Persistent Threats)," *InfoSec Resources*, 2018. [Online]. Available: https://resources.infosecinstitute.com/category/enterprise/phishing/phishing-as-an-attack-vector/phishing-apts-advanced-persistent-threats

[9] G. Singh, "Phishing & a live technical analysis," *SSRN Electron. J.*, 2017. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2940415#.

[10] K. Jansson and R. von Solms, "Phishing for phishing awareness," *Behav. Inf. Technol.*, vol. 32, no. 6, pp. 584–593, 2013. [Online]. Available: 10.1080/0144929x.2011.632650.

[11] T. Nikolaos, V. Nikos and M. Alexios, "Browser blacklists: The utopia of phishing protection," in *Proc. E-Business Telecommun*, 2014, pp. 278–293.

[12] W. Khan, M. Khan, F. Bin Muhaya, M. Aalsalem, and H. Chao, "A comprehensive study of email spam botnet detection," *IEEE Commun. Surv. Tuts.*, vol. 17, no. 4, pp. 2271–2295, 2015.

[13] S. Jeeva and E. Rajsingh, "Phishing URL detection-based feature selection to classifiers," *Int. J. Electron. Secur. Dig. Forensics*, vol. 9, no. 2, 2017, Art. no. 116.

[14] J. Chaudhry and R. Rittenhouse, "Phishing: Classification and countermeasures," in *Proc. Int. Conf. Multimedia*, 2016, pp. 28–31.

[15] H. Che, Q. Liu, and L. Zou, "A content-based phishing email detection method," in *Proc. IEEE Int. Conf. Softw. Quality Rel. Secur. Companion*, 2017, pp. 415–422.

[16] C. Tan, K. Chiew, K. Wong, and S. Sze, "PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder," *Decis. Support Syst.*, vol. 88, pp. 18–27, 2016.

[17] H. Shahriar and M. Zulkernine, "Trustworthiness testing of phishing websites: A behavior model-based approach," *Future Gener. Comput. Syst.*, vol. 28, no. 8, pp. 1258–1271, 2012.

[18] A. Ferreira and G. Lenzini, "An analysis of social engineering principles in effective phishing," in *Proc. Workshop Socio-Techn. Aspects Secur. Trust Within IEEE Comput. Secur. Foundations Symp.*, 2015, pp. 9–16.

[19] T. Spears, "Phishing for phools: The economics of manipulation & deception," *Quant. Finance*, vol. 17, no. 2, pp. 165–167, 2016.

[20] C. Konradt, A. Schilling, and B. Werners, "Phishing: An economic analysis of cybercrime perpetrators," *Comput. Secur.*, vol. 58, pp. 39–46, 2016. [Online]. Available: 10.1016/j.cose.2015.12.001

[21] N. Safa, M. Sookhak, R. Von Solms, S. Furnell, N. Ghani, and T. Herawan, "Information security conscious care behaviour formation in organizations," *Comput. Secur.*, vol. 53, pp. 65–78, 2015.

[22] J. Kang and D. Lee, "Advanced white list approach for preventing access to phishing sites," in *Proc. Int. Conf. Convergence Inf. Technol.*, 2007, pp. 491–496.

[23] A. Saeed, N. Dario, and X. Wang, "A comparison of machine learning techniques for phishing detection," in *Anti-Phishing Working Groups Ecrime Res. Summit*, 2007, pp. 60–69.

[24] S. Rawal, B. Rawal, A. Shaheen and S. Malik, "Phishing detection in e-mails using machine learning," *Int. J. Appl. Inf. Syst.*, vol. 12, no. 7, pp. 21–24, 2017.

[25] V. Gandhi and P. Kumar, "A Study on phishing: Preventions and anti-phishing solutions," *Int. J. Sci. Res.*, vol. 1, no. 2, pp. 68–69, 2012.

[26] J. Hong, "The state of phishing attacks," *Commun. ACM*, vol. 55, no. 1, 2012, Art. no. 74.

[27] B. Gupta, A. Tewari, A. Jain, and D. Agrawal, "Fighting against phishing attacks: State of the art and future challenges," *Neural Comput. Appl.*, vol. 28, no. 12, pp. 3629–3654, 2016.

[28] D. Komashinskiy, "An approach to detect malicious documents based on Data Mining techniques," *SPIIRAS Proceed.*, vol. 3, no. 26, 2014, Art. no. 126.

[29] N. Nissim, A. Cohen, C. Glezer, and Y. Elovici, "Detection of malicious PDF files and directions for enhancements: A state-of-the art survey," *Comput. Secur.*, vol. 48, pp. 246–266, 2015.

[30] X. Han, N. Kheir and D. Balzarotti, "PhishEye: Live monitoring of sandboxed phishing kits," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 1402–1413.

[31] Y. Cao, W. Han, and Y. Le, "Anti-phishing based on automated individual white-list," in *Proc. Workshop Digit. Identity Manage.*, 2008, pp. 51–60.

[32] A. Jain and B. Gupta, "A novel approach to protect against phishing attacks at client side using auto-updated white-list," *EURASIP J. Inf. Secur.*, vol. 2016, no. 1, 2016, Art. no. 9.

[33] G. Ramesh, I. Krishnamurthi, and K. Kumar, "An efficacious method for detecting phishing webpages through target domain identification," *Decis. Support Syst.*, vol. 61, pp. 12–22, 2014.

[34] S. Marchal, J. François, and R. State, "Proactive discovery of phishing related domain names," in *Proc. Int. Conf. Res. Attacks*, 2012, pp. 190–209.

[35] P. Tiwari and R. R. Singh, "Machine learning based phishing website detection system," *Int. J. Eng. Res.*, vol. 4, no. 12, pp. 172–174, 2015. [Online]. Available: 10.17577/ijertv4is120262.

[36] A. Jain and B. Gupta, "A machine learning based approach for phishing detection using hyperlinks information," *J. Ambient Intell. Humanized Comput.*, vol. 10, pp. 2015–2028, 2018.

[37] A. Jain and B. Gupta, "Comparative analysis of features based machine learning approaches for phishing detection," in *Proc. Int. Conf. Comput. Sustain. Global Develop.*, 2016, pp. 2125–2130.

[38] N. Abdelhamid, F. Thabtah, and H. Abdel-jaber, "Phishing detection: A recent intelligent machine learning comparison based on models content and features," in *Proc. IEEE Int. Conf. Intell. Secur. Inf.*, 2017, pp. 72–77.

[39] Y. Du and F. Xue, "Research of the anti-phishing technology based on e-mail extraction and analysis," in *Proc. Int. Conf. Inf. Sci. Cloud Comput. Companion*, 2014, pp. 60–65.

[40] T. Peng, I. Harris, and Y. Sawa, "Detecting phishing attacks using natural language processing and machine learning," in *Proc. IEEE Int. Conf. Semantic Comput.*, 2018, pp. 300–301.

[41] A. Bolton and C. Anderson-Cook, "APT malware static trace analysis through bigrams and graph edit distance," *Statist. Anal. Data Mining: ASA Data Sci. J.*, vol. 10, no. 3, pp. 182–193, 2017.

[42] P. Lakshmi, "Different similarity measures for text classification using KNN," *IOSR J. Comput. Eng.*, vol. 5, no. 6, pp. 30–36, 2012.
[43] A. Suryavanshi, "A survey paper on modified approach for kmeans algorithm," *Int. J. Emerg. Trends Sci. Technol.*, vol. 4, no. 3, pp. 17–34, 2016.
[44] K. Greff, R. Srivastava, J. Koutnik, B. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

Qi Li received the PhD degree in computer science and technology from the Beijing University of Posts and Telecommunications, China, in 2010. She is currently an associate professor with the Information Security Center, State Key Laboratory of Networking and Switching Technology, School of Computer Science, Beijing University of Posts and Telecommunications, China. Her current research focuses on information systems and software.

Mingyu Cheng received the BS degree in information security from the Xi'an University of Posts and Telecommunications, China, in 2019. He is currently working toward the MS degree in information security in the Beijing University of Posts and Telecommunications. His research interests include software security and data analysis.

Junfeng Wang received the MS degree in computer application technology from the Chongqing University of Posts and Telecommunications, Chongqing, in 2001, and the PhD degree in computer science from the University of Electronic Science and Technology of China, Chengdu, in 2004. From July 2004 to August 2006, he held a postdoctoral position in Institute of Software, Chinese Academy of Sciences. From August 2006, he is with the College of Computer Science and the School of Aeronautics & Astronautics, Sichuan University as a professor. His recent research interests include network and information security, spatial information networks and data mining.

Bowen Sun received the MS degree in information security from the Beijing University of Post and Telecommunications, Beijing, China, in 2019. He is currently a research assistant with CNITSEC. His recent research interests include malware analysis and machine learning.