IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 27, 2019, accepted April 20, 2019, date of publication April 29, 2019, date of current version May 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913705

Phishing Email Detection Using Improved RCNN Model With Multilevel Vectors and Attention Mechanism

YONG FANG, CHENG ZHANG, CHENG HUANG, LIANG LIU, AND YUE YANG
College of Cybersecurity, Sichuan University, Chengdu 610065, China

Corresponding author: Cheng Huang (opcodesec@gmail.com)

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 20826041B4249 and Grant 20826041B4252, and in part by the Sichuan University Postdoc Research Foundation under Grant 19XJ0002.

ABSTRACT The phishing email is one of the significant threats in the world today and has caused tremendous financial losses. Although the methods of confrontation are continually being updated, the results of those methods are not very satisfactory at present. Moreover, phishing emails are growing at an alarming rate in recent years. Therefore, more effective phishing detection technology is needed to curb the threat of phishing emails. In this paper, we first analyzed the email structure. Then, based on an improved recurrent convolutional neural networks (RCNN) model with multilevel vectors and attention mechanism, we proposed a new phishing email detection model named THEMIS, which is used to model emails at the email header, the email body, the character level, and the word level simultaneously. To evaluate the effectiveness of THEMIS, we use an unbalanced dataset that has realistic ratios of phishing and legitimate emails. The experimental results show that the overall accuracy of THEMIS reaches 99.848%. Meanwhile, the false positive rate (FPR) is 0.043%. High accuracy and low FPR ensure that the filter can identify phishing emails with high probability and filter out legitimate emails as little as possible. This promising result is superior to the existing detection methods and verifies the effectiveness of THEMIS in detecting phishing emails.

INDEX TERMS Email, phishing detection, classification, RCNN, attention.

I. INTRODUCTION

The rapid development of Internet technologies has immensely changed on-line users' experience, while security issues are also getting more overwhelming. The current situation is that new threats may not only cause severe damage to customers' computers but also aim to steal their money and identity. Among these threats, phishing is a noteworthy one and is a criminal activity that uses social engineering and technology to steal a victim's identity data and account information. According to a report from the Anti-Phishing Working Group (APWG), the number of phishing detections in the first quarter of 2018 increased by 46% compared with the fourth quarter of 2017 [1]. According to the striking data, it is clear that phishing has shown an apparent upward trend in recent years. Similarly, the harm caused by phishing can be imagined as well.

The associate editor coordinating the review of this manuscript and approving it for publication was Gaurav Somani.

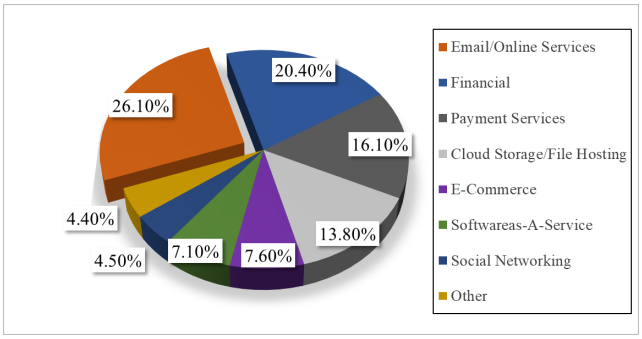


FIGURE 1. The industries targeted by phishing of *2018 phishing trends&intelligence report.*

As shown in Fig.1, the report from PhishLabs notes that email and online services overtook financial institutions as the top phishing target [2]. For phishing, the most widely used and influential mean is the phishing email. Phishing email refers to an attacker using a fake email to trick the recipient into returning information such as an account

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

password to a designated recipient. Additionally, it may be used to trick recipients into entering special web pages, which are usually disguised as real web pages, such as a bank's web page, to convince users to enter sensitive information such as a credit card or bank card number and password. Although the attack of phishing email seems simple, its harm is immense. In the United States alone, phishing emails are expected to bring a loss of 500 million dollars per year [3]. According to the APWG, the number of phishing emails increased from 68,270 in 2014 to 106,421 in 2015, and the number of different phishing emails reported from January to June 2017 was approximately 100,000 [4], [5]. In addition, Gartner's report notes that the number of users who have ever received phishing emails has reached a total of 109 billion [6]. Microsoft analyzes and scans over 470 billion emails in Office 365 every month to find phishing and malware. From January to December 2018, the proportion of inbound emails that were phishing emails increased by 250% [7]. Great harm and strong growth momentum have forced people to pay attention to phishing emails. Therefore, many detection methods for phishing emails have been proposed.

Various techniques for detecting phishing emails are mentioned in the literature. In the entire technology development process, there are mainly three types of technical methods including blacklist mechanisms, classification algorithms based on machine learning and based on deep learning. From previous work, the existing detection methods based on the blacklist mechanism mainly rely on people's identification and reporting of phishing links requiring a large amount of manpower and time. However, applying artificial intelligence (AI) to the detection method based on a machine learning classification algorithm requires feature engineering to manually find representative features that are not conducive to the migration of application scenarios. Moreover, the current detection method based on deep learning is limited to word embedding in the content representation of the email. These methods directly transferred natural language processing (NLP) and deep learning technology, ignoring the specificity of phishing email detection so that the results were not ideal [8], [9].

Given the methods mentioned above and the corresponding problems, we set to study phishing email detection systematically based on deep learning. Specifically, this paper makes the following contributions:

1) With respect to the particularity of the email text, we analyze the email structure, and mine the text features from four more detailed parts: the email header, the email body, the word-level, and the char-level.
2) The RCNN model is improved by using the Bidirectional Long Short-Term Memory (Bi-LSTM). Then, the email is modelled from multiple levels using an improved RCNN model. Noise is introduced as little as possible, and the context information of the email can be better captured.
3) The attention mechanism is applied between the email header and the email body, and different weights are respectively assigned to the two parts so that the model can focus on more different and more useful information from the email header and the email body.
4) The THEMIS model proposed in this paper performs well on an unbalanced dataset. The accuracy achieves 99.848%, and all evaluation metrics of THEMIS are superior to the existing detection technologies.

The remainder of the paper is structured as follow. In the next section, we will discuss the related work. Section III further discusses the proposed method. Section IV presents the experimental results and gives the analysis. At last, Section V concludes the paper and proposes the prospect.

II. RELATED WORK

With the emergence of email, the convenience of communication has led to the problem of massive spam, especially phishing attacks through email. Various anti-phishing technologies have been proposed to solve the problem of phishing attacks. Sheng *et al*. [10] studied the effectiveness of phishing blacklists. Blacklists mainly include sender blacklists and link blacklists. This detection method extracts the sender's address and link address in the message and checks whether it is in the blacklist to distinguish whether the email is a phishing email. The update of a blacklist is usually reported by users, and whether it is a phishing website or not is manually identified. At present, the two well-known phishing websites are PhishTank and OpenPhish. To some extent, the perfection of the blacklist determines the effectiveness of this method based on the blacklist mechanism for phishing email detection.

With the development of AI, phishing email detection has also entered the era of machine learning. In particular, the combination of NLP and machine learning has played a significant role in phishing email detection. Semantic features [11], syntax feature [12], and contextual features [13] previously have been used in this area. Vazhayil *et al*. [14] started from the most basic machine learning methods and used decision trees, logistic regression, random forests, and SVM combined with supervised classification to detect phishing emails. Hamid and Abawajy [15] proposed a hybrid feature selection method that combines content and behavior. The detection method for phishing emails using machine learning mainly uses marked phishing emails and legitimate emails to train the classification algorithm in the machine learning algorithm to obtain the classifier model for email classification. Bergholz *et al*. [16] put forward a series of features that are classified into three sets: basic features [16], [17], latent topic model features [16], [18], [19], and dynamic Markov chain features [16], [17], [19]. The basic features are what can be extracted directly from an email without extra processing. Topic model features are potential features that cannot be observed in an email. Specifically, it is mainly some words that are related to each other and may appear together. Dynamic Markov chain features are text features based on the bag-of-words; that is, the goal of capturing the probability of an email belonging to a specific

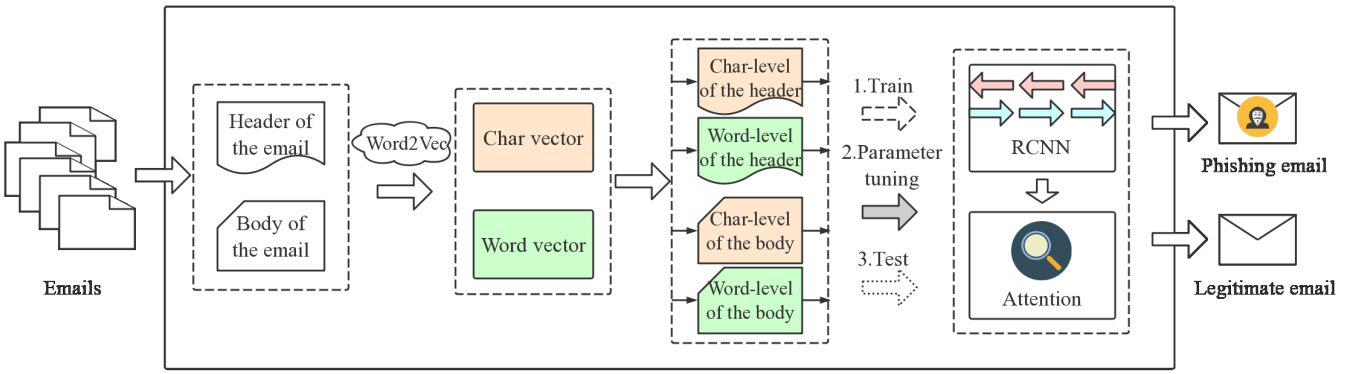


FIGURE 2. The framework for classifying phishing emails and legitimate emails in this paper.

category is achieved by modeling each type of message content. A drawback of NLP based on machine learning in phishing email detection is that it has been based on an email's surface level text, rather than deep semantics. Therefore, the use of synonyms, different sentence construction, and other differences are difficult to find by NLP based on machine learning [20]. Also, the machine learning method mainly relies on feature engineering to generate features representing emails and performs tasks through these features. It is obvious that both blacklisting and feature engineering need to be completed manually and require a large amount of labor and professionals with domain expertise, which limits the performance of detection. To solve the problems that exist in the first two methods, the following studies focus on deep learning techniques.

Deep learning has been well-represented in many NLP tasks, including text categorization [21], information extraction [22], and machine translation [23]. It can also automatically generate effective features from emails to detect phishing emails, thus avoiding the manual extraction of email features. Therefore, the focus of using deep learning for phishing email detection is on characterizing the email text information more completely and comprehensively. Repke and Krestel [24] brought back structure to free text email conversations with deep learning and word embedding. Although this work is not to detect phishing emails, it is still instructive for us to use deep learning and word embedding to process emails. Hiransha *et al*. [8] proposed using Keras [25] word embedding and convolutional neural network (CNN) to build a phishing email detection model. There are other deep learning algorithms that are being used, such as the Deep Belief Network (DBN) and the Recurrent Neural Network (RNN) [26]–[28]. At present, these deep learning methods for phishing email detection are simply transferring NLP technology to phishing email detection, ignoring the differences between phishing email detection and other targets. Context information is ignored to some extent. All of these have caused limitations for the development of phishing email detection.

Lai *et al*. [29] proposed RCNN for text classification in 2015 and tested it on four datasets including the 20Newsgroups Dataset [30], the Fudan set [31], the ACL Anthology Network [32], and the Stanford Sentiment Treebank [33], all of which were better than baseline. RCNN can produce as little noise as possible and has high performance in complicated sequence classification tasks [34]. Based on the reason that email is also text with complex content, the THEMIS model of this paper is built on RCNN.

Bahdanau *et al*. [23] proposed the attention mechanism in machine translation. Luong *et al*. [35] applied an attention mechanism to neutral machine translation (NMT). The attention mechanism is very useful in many fields and can improve image classification [36], automatic image captioning [37] and machine translation [38]. Yang *et al*. [39] explored a hierarchical attention mechanism for document classification and achieved good results in 2016. It is clear that the attention mechanism also has a positive impact on text classification. To some extent, this shows the correctness of our THEMIS model using the attention mechanism.

III. PROPOSED APPROACH

In this paper, emails are divided into two categories, legitimate emails and phishing emails. Naturally, the detection for phishing emails is also a binary classification problem. We mathematize the problem and split an email into two parts, the header and the body. We define a binary variable y to represent the attributes of an email; that is, $y = 1$ means that the email is a phishing email and $y = 0$ means that the email is legitimate. In other words, y is the label of an email. We follow the following steps to determine whether the email is a phishing email. To begin this process, we calculate the probability that the email is a phishing email, that is, $P(y = 1)$. Then, the probability value is compared with the classification threshold, and if it is greater than the classification threshold, it is judged as a phishing email. Our goal is to detect whether the target email is legitimate or phishing quickly and accurately. In this section, we will present the details of our proposed model.

A. OVERVIEW

Fig.2 shows our framework for classifying phishing emails and legitimate emails. First, because the content of an email

is very irregular, we need to simply process the email dataset– replace and delete the extra spaces and digital gibberish in the text. The email is divided into multiple levels: the char-level and the word-level of the email header as well as the char-level and the word-level of the email body. Then, Word2Vec [40] is used to train and obtain the sequences of vectors. Next, we divide the data into two parts, one as a training-validation set and the other as a testing set. We input a part of the training-validation set into our model and train it to obtain the classifier. Additionally, the other part of the training-validation set is used to carry out the super-parameter selection experiment on the classifier to obtain the best classification threshold. Finally, the testing set is used to test the determined classifier model to verify the function.

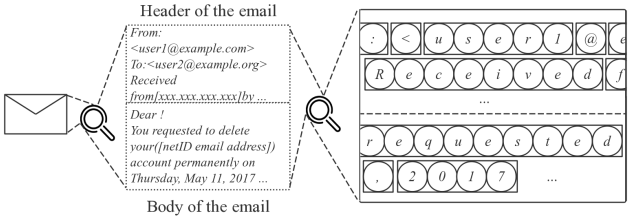


FIGURE 3. The analysis of email structure. In the figure, a circle represents a character, and a rectangle represents a word. A rectangle is filled with an indefinite number of circles, indicating that the word consists of an indefinite number of characters.

B. MULTILEVEL EMBEDDING

The manual extraction of features is avoided because we adopted the deep learning method. To achieve the best result for deep learning, we need completer and more sufficient information to characterize all important information about the input data. For phishing email detection, the input data are the text content of emails. How can we better express the text content of the email in the form of vectorization? As shown in Fig.3, an email text can be thought of as consisting of two parts: the header and the body. The body is the core of the information conveyed by an email. Because this part is controlled by people entirely, the email body is very random. However, in order to achieve the purpose of attack, there is often some fascinating or warning information in the body of phishing emails, which is different from legitimate emails in deep semantics. This information can make use of people's psychological weakness, attract the attention of victims as much as possible, and improve the possibility of victims visiting phishing websites given in phishing emails. For example, most phishing emails are disguised as banks to inform users that there are abnormalities in their account numbers. Besides, the header is always located above the email body and contains the particular routing information of the email. Compared with the email body, the header is more regular. The email header consists of a series of key-value pairs, where the keys are fixed contents such as *From*, *To*, and *Subject*. When an attacker forges the identity of the sender to send email, in order to deceive the victim, a small part of the header content will be forged. However, there are still a large number of headers that cannot be modified. Therefore, we can dig deep-level features from the relationship between the contents of the whole email header. For example, whether the domain name of *Message-Id* in the email matches the domain name of the sender. In short, there are significant differences between the email header and the email body. Naturally, for the detection of phishing emails, we should start with the email header and the email body, respectively. This enables our model to focus on the deep features of the email body or the email header separately and simultaneously under the limited input length of the model, which is beneficial to our detection.

As can also be seen from Fig.3, not only email but also all text content has two most basic constituent units: char-level basic units and word-level basic units. Characters can form all kinds of words. In addition, words can also form all kinds of sentences. Therefore, we should also start with the most basic units of text for the representation of an email. Most of the existing papers focus only on the word-level vector. However, the char-level vector can better concentrate on spelling mistakes, personal spelling habits, uppercase words, and lowercase words [20]. All of these are difficult to achieve with the word-level vector alone. As for the email text, the content of the email body is very individualized, for example, the spelling habits, the uppercase and lowercase words, and some spelling mistakes are likely to occur. Hence, we can take advantage of the individual email writing traits to distinguish phishing emails from legitimate emails. Moreover, the values in the email header are not all words, and there are also some fixed character sequences with specific combinations, such as a domain name. The fixed character sequences with specific combinations are more accessible to learn by using the char-level vector. Therefore, in addition to using the word-level vector, the char-level vector is used. In this paper, the char-level representation and word-level representation of an email are obtained using Word2Vec. It is a popular model proposed by Mikalov, which is used to generate word embedding on text data [41]. It reproduces the linguistic context of words by training the shallow two-layer architectures. The input of Word2vec model is a huge corpus, and the generated outputs are vectors of some specified dimensions. Each unique word (or character) in the corpus has a corresponding vector associated with it, which reflects the context of the word (or character) [42]. This makes learning the representation of words (or characters) significantly faster than previous methods.

In summary, to represent the email, we can describe it from multiple levels of the char-level of the email header, char-level of the email body, word-level of the email header and word-level of the email body to better reflect all information contained in the email. The char-level vector embedding model and the word-level vector embedding model are obtained through Word2Vec tool training. Enter the email into these two models and the results are the vector sequences of the char-level email header, the word-level email

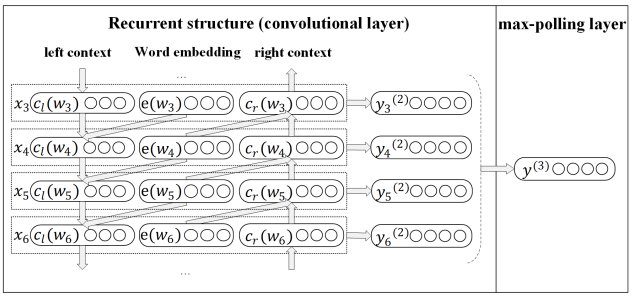


FIGURE 4. The main structure of RCNN proposed in *recurrent convolutional neural networks for text classification* by S. Lai *et al.*

header, the char-level email body and the word-level email body.

C. RECURRENT CONVOLUTIONAL NEURAL NETWORKS AND ATTENTION MECHANISM

RCNN is a new deep learning algorithm proposed by Lai *et al.* in 2015 [29]. The structure of RCNN is shown in Fig.4. It defines w_{i-1} and w_{i+1} are the previous word and the next word of w_i. α can be l or r which means left and right respectively. β can be w_i, w_{i-1}, w_{i+1}, *etc*. $c_\alpha(\beta)$ is the left or right context of the word β, which is a dense vector with $|c|$ real value element. f is a non-linear activation function. $W^{(\alpha)}$ is the matrix that transforms the hidden layer into the next hidden layer. $W^{(s\alpha)}$ is a matrix that is used to combine the semantics of the current word with the next word's left or right context. $e(\beta)$ is the word embedding of the word β, which is a dense vector with $|e|$ real value elements. Through Equ.(1) and Equ.(2), we get the left-side context vector $c_l(w_i)$ and the right-side context vector $c_r(w_i)$ of word w_i. Through Equ.(3), the representation x_i of such the word w_i becomes a concatenation of the word vector, the forward and backward context vector. It can be seen that the forward and backward RNNs are used to get the representation of the forward and backward context of each word. Next, as shown in Equ.(4), $b^{(2)}$ is the bias vector. Also, the linear transformation and the *tanh* activation function are applied to x_i together. We obtain result $y_i^{(2)}$, and send the result to the next layer. What the max-pooling layer needs to do is to get the representation $y^{(3)}$ of all words through Equ.(5). The *max* is an element-wise function. Finally, the output layer is similar to other general neural networks.

$$c_l(w_i) = f(W^{(l)}c_l(w_{i-1}) + W^{(sl)}e(w_{i-1})) \quad (1)$$

$$c_r(w_i) = f(W^{(r)}c_r(w_{i+1}) + W^{(sr)}e(w_{i+1})) \quad (2)$$

$$x_i = [c_l(w_i); e(w_i); c_r(w_i)] \quad (3)$$

$$y_i^{(2)} = tanh(W^{(2)}x_i + b^{(2)}) \quad (4)$$

$$\boldsymbol{y}^{(3)} = \max_{i=1}^{n} \boldsymbol{y}_i^{(2)} \quad (5)$$

Through the experimental comparison in the paper by S. Lai *et al.*, about capturing contextual information, the RCNN model outperforms the CNN that uses convolutional layer and the RNN that uses semantic composition under the constructed textual tree. The recurrent structure in RCNN can preserve longer contextual information and introduce less noise. The purpose of phishing emails is rather clear. No matter how disguised the attacker is, his ultimate goal of attacking the victim will not be changed, and it will be exposed in the text expression process. The RCNN model is used to analyze the email text–captures contextual information with the recurrent structure and constructs the representation of text using a convolutional neural network. The RCNN model can discover the deep-level features of phishing email different from legitimate email, and then achieve the goal of detecting phishing emails.

In the RCNN model, S. Lai *et al.* use the bidirectional Recurrent Neural Network (BRNN) to capture the contexts. However, the RNN has a long-term dependency problem, which may further cause the gradient exploding and vanishing problems. To address the problems in RNN, variants of RNN have been proposed, one of which typically is the Long Short-Term Memory (LSTM) [43]. The LSTM uses gates on the input and the recurrent input to control the state as well as output at a different time to achieve the purpose of learning such "long-term dependence" information. In this paper, the RCNN model is used to learn the text features of email, which is a long text sequence. As mentioned earlier, because the RNN structure in RCNN model cannot learn such "long-term dependence" information from emails, so we improve the RCNN model. The BRNN in the original RCNN model is replaced, and the Bi-LSTM is used to obtain the left and right semantic information of a certain location. Combined with its embedding, it finally forms a triple. That is, each word can be represented by such a triple.

The attention mechanism in deep learning is substantially similar to the selective visual attention mechanism of human beings, and the core goal is to select information more critical to the goal of the current task from much information. The improved RCNN and attention mechanism are combined to form an improved RCNN-Attention model. The structural diagram is shown in Fig.5. After inputting the vectorized representation of data, first, Bi-LSTM is used to obtain the left and right semantic information of a certain location with its embedding information, thus forming a triple; that is, each word can be represented by such a triple. Then, a dense layer with a *tanh* activation function is used to map such triples to a specified dimension, and longitudinal max polling is adopted to obtain the final representation. Finally, the attention mechanism is adopted.

D. THEMIS MODEL

Based on the multilevel embedding and improved RCNN-Attention model mentioned earlier, we formally put forward the THEMIS model, a phishing email detection model, by combining these two parts. The frame is shown in Fig.6. We vectorize the email according to its text structure: header, body, characters, and words, using Word2Vec to output char-level embedding and word-level embedding. We combine the text structure of the email into the char-level of the email

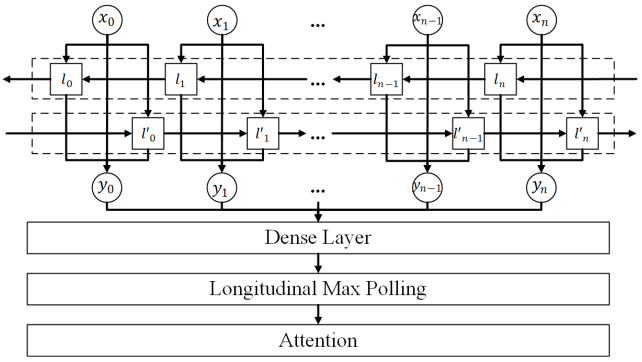


FIGURE 5. Improved RCNN-Attention model. Two dashed boxes represent Bi-LSTM. $x_0, x_1, \ldots, x_{n-1}, x_n$ represent the input of basic units. $l_0, l_1, \ldots, l_{n-1}, l_n$ and $l'_0, l'_1, \ldots, l'_{n-1}, l'_n$ represent the acquired left context and right context respectively. $y_0, y_1, \ldots, y_{n-1}, y_n$ are triples formed by the left context and right context of a certain position and its own embedding vector.

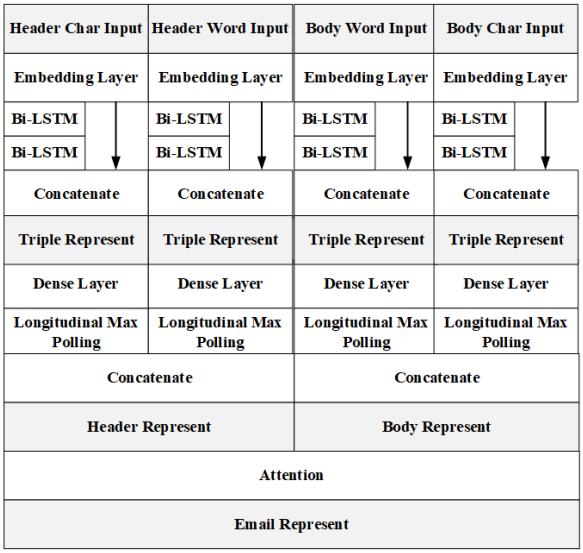


FIGURE 6. The structure of the model named THEMIS.

header, word-level of the email header, char-level of the email body and word-level of the email body, and we input them into the embedding layer and Bi-LSTM. The result obtained is then connected to the result obtained by the embedding layer to form a triple representation. Then, the triples are sequentially inputted into the dense layer and longitudinal max polling. The results are connected according to the header and the body to obtain the representation of the email header and the representation of the email body. As mentioned in subsection B, different features of email can be mined from the email header and the email body, respectively. The email body mainly contains semantic features, and the email header mainly contains routing features and a small part of semantic features (mainly for the *subject*). Sometimes, in the body of an email, most phishing emails are nearly identical to the legitimate email. In this case, we should give low weight to the content of the email body to ensure the accuracy of the phishing email detection results. It can be seen that under certain circumstances, the email header content and the email body content have varying degrees of impact on phishing email detection. Based on the situation as mentioned above, the attention mechanism is used to obtain the weighted sum of the email header and the email body, which is the representation of the whole final email.

It can be seen that we only apply the attention mechanism between the representation of the email header and the representation of the email body, not between the four parts–the char-level of the email header, word-level of the email header, char-level of the email body and word-level of the email body. The reason is that char-level representation and word-level representation can represent semantics from different levels of the email text, so concatenate is directly used between char-level representation and word-level representation of the email header and the email body, which can retain more information than using the attention mechanism.

E. CLASSIFICATION THRESHOLD MOVING

In the binary classification experiment, when we classify a sample x, we are actually comparing the predicted probability value y with the classification threshold value p. For example, it is usually determined as a positive example when $y > p$; otherwise, it is determined as a negative example. y actually expressed the possibility of a positive example. $y/(1-y)$ represents the ratio of the two kinds of possibilities, which is also called the odds ratio. Assuming that the number of positive examples is m and the number of negative examples is n, then the observed odds ratio is m/n. Since we usually assume that the training set is an unbiased sampling of the real sample population, as shown in Equ.(6), the observation odds ratio represents the real odds ratio.

$$\frac{y}{1-y} = \frac{m}{n} \tag{6}$$

That is, the value of the classification threshold p is equal to:

$$y = \frac{m}{(m+n)} \tag{7}$$

As can be obtained from Equ.(7), in the case of class-balance ($m = n$), the classification threshold is $p = 0.5$. There is a problem of class-imbalance ($m \neq n$) in our dataset. Therefore, unlike other class-balance experiments that use the probability of 0.5 as the classification threshold, we should move the classification threshold in our experiment. We can determine the classification threshold by adjusting the super-parameter on the validation set to achieve the optimal performance of the model.

IV. EXPERIMENTAL AND EVALUATION

The computer used in running all the experiments in this paper is a PC, having a 3.60 GHz of CPU, 16 GB of RAM, GTX 1060 of GPU. The THEMIS model is implemented by TensorFlow [44] and Keras.

A. DATASET

The experimental data come from the First Security and Privacy Analytics Anti-Phishing Shared Task (IWSPA-AP 2018) [45]. The email data sources of this dataset are diversified. The sources of the legitimate email include email collections from Wikileaks archives, such as the Democratic National Committee, Hacking Team, Sony emails, *etc*. There are also selected emails from the Enron Dataset [46] and SpamAssassin. As for the phishing emails, they mainly come from the Information Technology (IT) departments of different universities, the popular Nazario's phishing corpora [47], and synthetic emails created by organizers using Dada engine [48], which is a system that generates text based on a pre-specified grammar. This dataset has been preprocessed to a certain extent, including replacing all the URLs with $<<$ *link* $>>$ or live phishing links from PhishTank, deleting any possible signs of the dataset source, and removing all base64 encoded text. Moreover, the dataset has been deleted emails that are too big (more than 1 MB) or too small (the threshold for removing smaller size emails varies with each dataset). The dataset has been divided into a training set and testing set. Both the training set and the testing set contain emails without header and emails with header. In this paper, we only focus on email data with the header. Due to the irrationality of the segmentation of the training set and the testing set in the original dataset, after merging the two datasets, the training-validation set and the testing set are re-divided. The dataset is divided by stratified random sampling; that is, random samples are taken from legitimate email and phishing email at the same proportion. This ensures that the two datasets used in training and testing phases are well represented. The dataset structure after division is shown in Table 1.

TABLE 1. The Details of dataset used in this paper.

Dataset	legitimate	phishing	Total
Training-validation set	5,447	699	6,146
Testing set	2,334	300	2,634
Total	7,781	999	8,780

Here, we do not divide the data set into a training set, a validation set and a testing set according to the traditional deep learning. Instead, the data set is divided into training-validation set and testing set. After that, we use 10-fold cross-validation on the training-validation set to train the model and choose the super-parameter.

B. EXPERIMENTAL STEPS

A series of preparatory work is needed before conducting the main experiment. In order to eliminate as much noise as possible, the data provider has attempted to remove the empty spacing that resulted from parsing the email body using an HTML parser, but the effect is not ideal [49]. So, we use the regular expression to match then delete blank lines and unnecessary spaces that still exist in the email body. To the greatest extent, we reduce noise and keep as much email information as possible. Due to the input of the model, A python library called *email* [50], which provides a standard parser that understands most email document structures, is used to extract the email headers from the processed data and then divide the email into two parts, namely the email header and the email body. In the email header part, we save each field as a key-value pair. At the same time, the email header part and the email body part are segmented by word segmentation and character segmentation. Email is different from other texts, and its content has some uniqueness. In the vectorization phase, it is necessary to train the word vector and character vector for email data. We use the Google Word2Vec tool to train with the dimension set to 256. Finally, we obtain the char-level vector embedding model and the word-level vector embedding model.

The char-level vector embedding model and the word-level vector embedding model obtained from the previous training are used to convert the email header part and the email body part into sequences of vectors. Taking full account of the data statistics and experimental resource constraints, we set the length of the four input sequences as 150, 80, 300 and 300, respectively, for the char-level sequence of the email header, the word-level sequence of the email header, the char-level sequence of the email body and the word-level sequence of the email body. Therefore, sequences shorter than the set value are filled with [0, 0]. At the same time, the sequence whose length exceeds the set value is truncated. Then the length of the four input sequences processed is a fixed value. As stated above, we use 10-fold cross-validation. The training-validation set in fixed-length data is used to train the classification model with 20 epochs. The current effect of the model is verified by using the training- validation set after each epoch of training. Details of the entire model implemented based on Keras and TensorFlow are shown in Fig.7. The first two layers are the input and embedding layers of the char-level sequence of the email header, the word-level sequence of the email header, the char-level sequence of the email body, and the word-level sequence of the email body. The third and fourth layers are to directly use the LSTM network and control *go_backwards* parameter values to realize four parallel Bi-LSTMs. The fifth to seventh layers realize the contents of Equ.(3), Equ.(4) and Equ.(5), respectively. The eighth layer and the ninth layer are used to connect the char-level sequence and the word-level sequence of the email header and the email body to form the final email header representation and the email body representation, respectively. The purpose of the tenth to seventeenth layers is to realize the attention mechanism between the email header and the email body, then form the final representation of the whole email. *Dropout* of the penultimate layer is to prevent overfitting. The *sigmoid* function is used as the activation function in the last layer for the reason of binary classification, and the final output of the model is obtained.

We select the model obtained from the previous epoch of training when the model validation index began to decline as

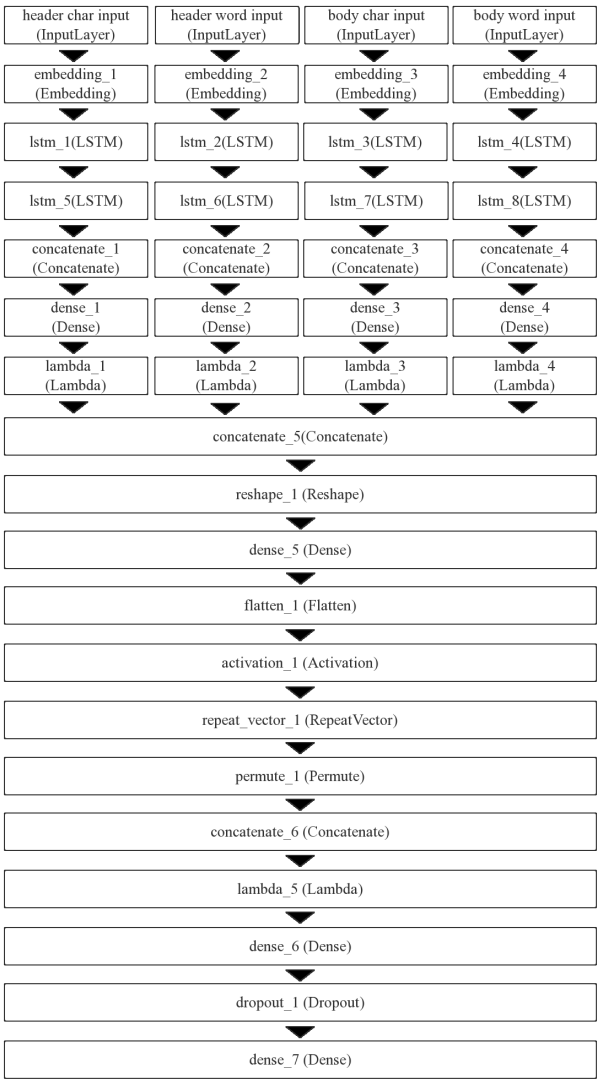


FIGURE 7. Details of model configuration implemented by code.

the basic model. Based on this model, we select the super-parameter. As mentioned earlier, our data have the problem of class-imbalance. We can obtain a classification threshold of 0.1138 from Equ.(7) and Table 1. That is, samples with an output probability greater than 0.1138 are determined to be a phishing email. Otherwise, they are determined to be a legitimate email. Then, we fine-tune the classification threshold on the training-validation set to achieve the best classification effect. According to the output of the model on the training-validation set, we draw the Precision-Recall (P-R) curve as shown in Fig.8. According to the definition of the P-R curve, when the curve intersects with $y = x$, the intersection point reaches the state of $Precision = Recall$ and that is when the model is best. This point is called the Break-Even Point (BEP). The classification threshold corresponding to the BEP is the best classification threshold.

When the classification threshold is 0.79829, the model is optimal. Therefore, we choose 0.79829 as the classification

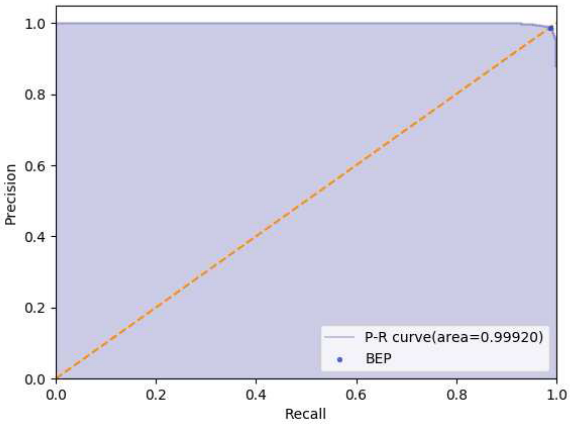


FIGURE 8. The P-R curve. The classification threshold corresponding to BEP is 0.79829.

threshold. Thus far, the model has been completely confirmed, and next, we evaluate the performance of the model on the testing set.

C. EVALUATION AND RESULT

We will input the testing set into the model that we have already obtained and obtain a series of indicators. These indicators are used to evaluate the performance of the model specifically.

Suppose that True Negative (TN) is the number of legitimate emails classified as legitimate, False Positive (FP) is the number of legitimate emails misclassified as phishing, False Negative (FN) is the number of phishing emails misclassified as legitimate, and True Positive (TP) is the number of phishing emails classified as phishing emails. Table 2 shows the classification confusion matrix.

TABLE 2. The classification confusion matrix.

Actual \ Predict	0 (legitimate)	1 (phishing)
0 (legitimate)	TN	FP
1 (phishing)	FN	TP

We consider the following aspects of testing the whole model.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{8}$$

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

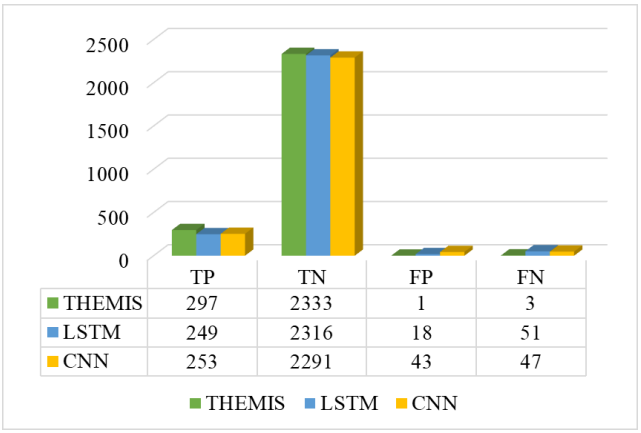
$$Recall = \frac{TP}{TP + FN} \tag{10}$$

$$F1 - score = 2 * \frac{precision * recall}{precision + recall} \tag{11}$$

$$FPR = \frac{FP}{FP + TN} \tag{12}$$

In order to make an objective evaluation, we implemented the methods as baseline proposed by Ra *et al.* [51], which

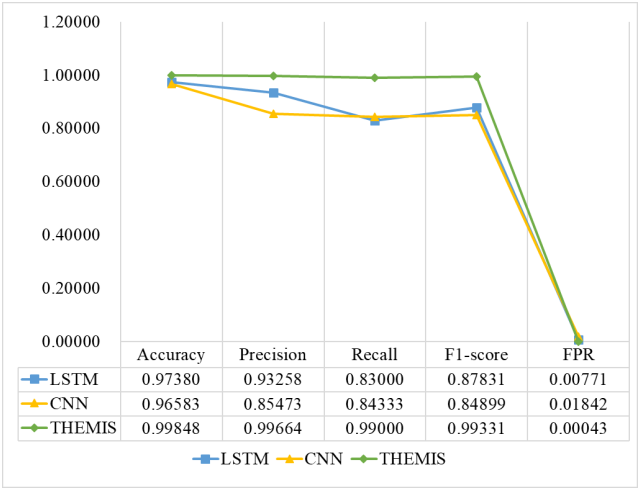
used CNN and LSTM algorithms. They are evaluated with the same data set used in THEMIS. The result summary is shown below.



	TP	TN	FP	FN
THEMIS	297	2333	1	3
LSTM	249	2316	18	51
CNN	253	2291	43	47

FIGURE 9. The confusion matrix of test results.

For the confusion matrix result, it is expected that the value of TP and TN is large, while the value of FP and FN is small. As can be seen from the result of the confusion matrix in Fig.9, our model has larger TP and TN, and smaller FP and FN, compared with the other two models. The result of the confusion matrix is limited to the number. When faced with a large amount of data such as this experiment, it is difficult to adequately measure the model's advantages and disadvantages only by evaluating it regarding number. Therefore, we use the result of the confusion matrix to calculate and obtain further evaluation indexes.



	Accuracy	Precision	Recall	F1-score	FPR
LSTM	0.97380	0.93258	0.83000	0.87831	0.00771
CNN	0.96583	0.85473	0.84333	0.84899	0.01842
THEMIS	0.99848	0.99664	0.99000	0.99331	0.00043

FIGURE 10. The summary of test results in terms of accuracy, precision, recall, F1-score and FPR.

As is indicated in Fig.10, the accuracy of THEMIS model is 99.848%, recall is 99.000%, precision is 99.664%, F1-score is 99.331%, and FPR is 0.043%. All the performances are better than the methods of CNN and LSTM. It is worth noting that the FPR represents the probability that the model will judge legitimate email as a phishing email, and a lower FPR is necessary for the email filtering system. Our model does that and reduces the risk of filtering out the legitimate email to a greater extent.

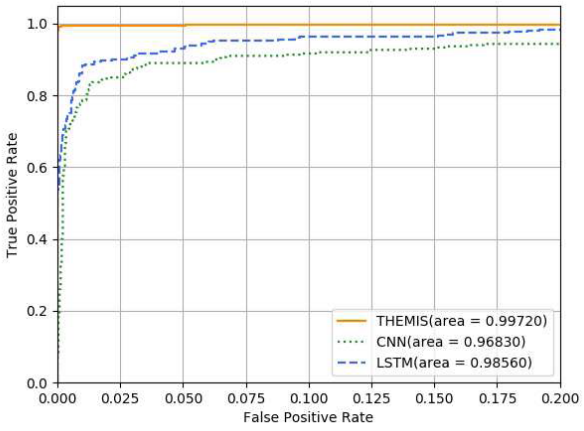


FIGURE 11. The ROC curve of THEMIS, CNN and LSTM.

The Receiver Operating Characteristic (ROC) curve is shown in Fig.11. The area under the curve (AUC) of our THEMIS is 99.720%, while the counterparts are 96.830% and 98.560%. We can see from the curve trend that our model has a stronger classification ability.

Although deep learning is not interpretable to some extent, we still try to analyze the experimental results by the misclassified emails. We try to divide emails of these four parts (TP, TN, FP and FN) from the confusion matrix of CNN into an email header and email body, then use the CNN model to detect the email header and email body, respectively. And We compare the results with the results obtained by using the whole email. As shown in (b) and (d) of Fig.12, the detection results using only email header and only email body are different for the same misclassified email. When only using the email header to detect phishing emails, the CNN model can correctly predict the most of results. But, why does CNN model make a wrong prediction when using the whole email (including the email header and the email body)? As can be seen by comparing (b) and (d) with (a) and (c) in Fig.12, the results of detection using the whole email are affected to some extent by the email body. In other words, the email body dominates the detection results of the whole email. As can be seen from (b) of Fig.12, the email headers of most phishing emails are determined as phishing emails, while the corresponding email body is opposite. Because the body of email determines the final test result, so the CNN model fails to accurately detect these phishing emails. We believe that the emails that CNN failed to detect belong to highly hidden phishing emails whose email body has extremely high similarity with the body of legitimate emails (the email header still has differences). In this case, we should give low weight on the email body and high weight on the email header to

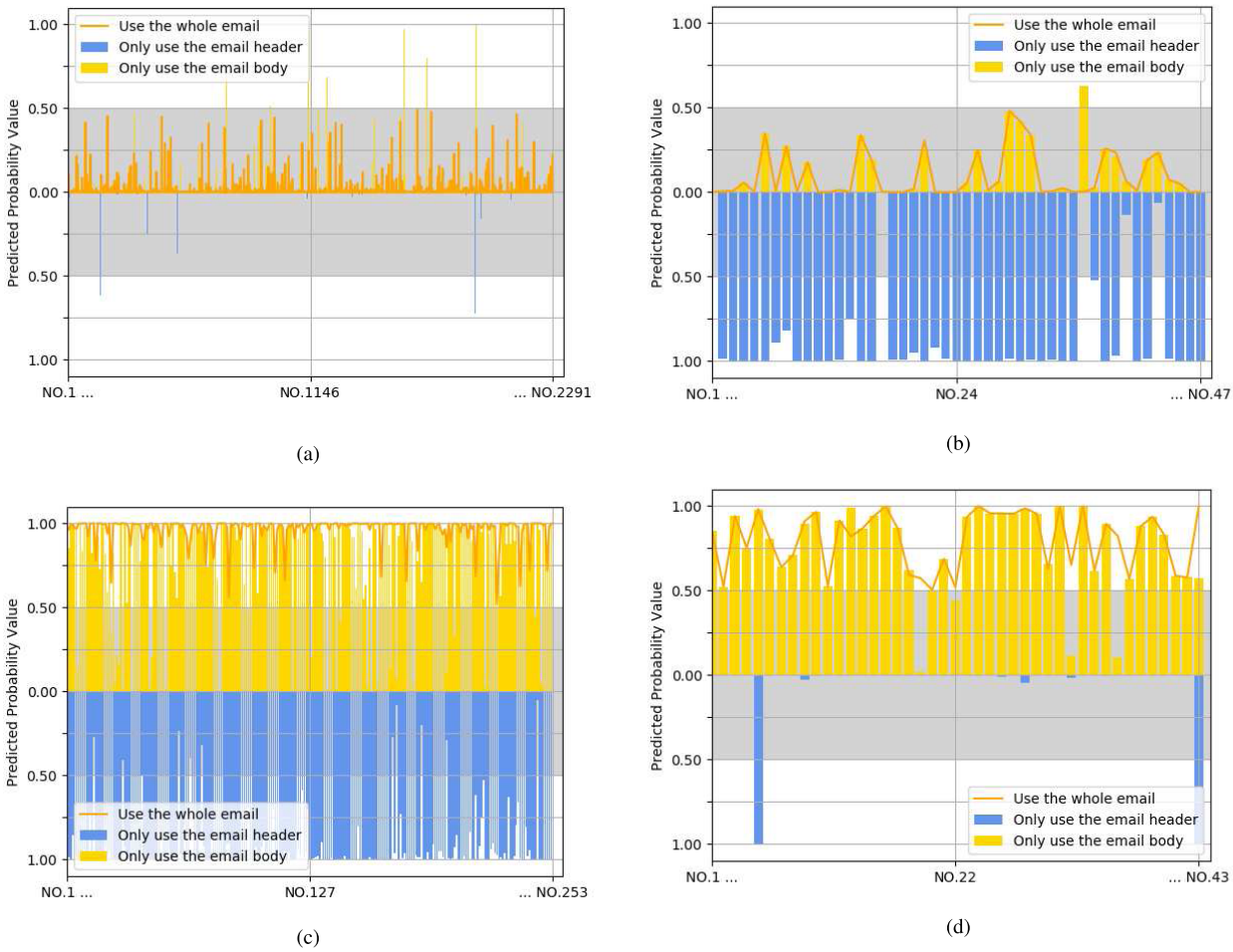


FIGURE 12. The prediction probabilities of using the CNN model to detect emails from the whole email, only email header, and only email body, respectively. The X-axis indicates the email number. The Y-axis represents the probability value predicted by the CNN model. The grey area represents legitimate, and other areas indicate phishing, that is if the probability value of a certain email falls in the grey area, it represents that the email is classified to be legitimate by the CNN model. Otherwise, the email is regarded as phishing. (*We draw the "only use email header", that is, the blue result, to the lower half plane to make the figure clear.* (a) Legitimate emails classified as legitimate (A total of 2,291 emails). (b) Phishing emails misclassified as legitimate (A total of 47 emails). (c) Phishing emails classified as phishing (A total of 253 emails). (d) Legitimate emails misclassified as phishing (A total of 43 emails).)

ensure accurate detection of the phishing email. When we use the attention mechanism between the email header and the email body to assign them different weights dynamically, through experiments, our model can accurately detect these misclassified emails in the case shown in (b) and (d) of Fig.12 because attention mechanism gives the email header a higher weight than the email body. This further illustrates the rationality of the attention mechanism in THEMIS model.

In terms of LSTM model, there are 51 phishing emails have not been detected. We find that these emails have a high degree of camouflage in both the email header and the email body, such as the sentence structure is more complicated and the difference in spelling habits is more obvious. In this case, it is tough to get ideal results if we detect phishing emails by shallow features. Relevant research notes that the convolution-based framework is more suitable for constructing the semantic representation of texts compared with RNN. And the recurrent structure in the RCNN captures contextual information better than the window-based structure in CNN [29]. Therefore, although the LSTM model can capture certain semantic information, it is difficult to accurately capture the deep semantic information of these emails, which RCNN can do. Moreover, we use not only word-level vector embedding but also char-level vector embedding, which makes the captured feature details of both the email header and the email body more comprehensive. As mentioned above, only through these improvements can our model perform significantly better than the LSTM model.

V. CONCLUSION AND FUTURE WORK

In this paper, we use a new deep learning model named THEMIS to detect phishing emails. The model employs an improved RCNN to model the email header and the email body at both the character level and the word level. Therefore, the noise is introduced into the model minimally. In the model, we use the attention mechanism in the header and

the body, making the model pay more attention to the more valuable information between them. We use the unbalanced dataset closer to the real-world situation to conduct experiments and evaluate the model. The THEMIS model obtains a promising result. Several experiments are performed to demonstrate the benefits of the proposed THEMIS model. For future work, we will focus on how to improve our model for detecting phishing emails with no email header and only an email body.

ACKNOWLEDGMENT

The authors would like to thank IWSPA-AP 2018 for providing experimental data in this research.

REFERENCES

[1] Anti-Phishing Working Group. (2018). *Phishing Activity Trends Report 1st Quarter 2018*. [Online]. Available: http://docs.apwg.org/Preports/apwg_trends_report_q1_2018.pdf

[2] PhishLabs. (2018). *2018 Phish Trends & Intelligence Report*. [Online]. Available: https://info.phishlabs.com/hubfs/2018%20PTI%20Report/PhishLabs%20Trend %20Report_2018-digital.pdf

[3] M. Nguyen, T. Nguyen, and T. H. Nguyen. (2018). "A deep learning model with hierarchical LSTMs and supervised attention for anti-phishing." [Online]. Available: https://arxiv.org/abs/1805.01554

[4] Anti-Phishing Working Group. (2016). *Phishing Activity Trends Report 4th Quarter 2016*. [Online]. Available: http://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf

[5] Anti-Phishing Working Group. (2015). *Phishing Activity Trends Report 1st-3rd Quarter 2015*. [Online]. Available: http://docs.apwg.org/Preports/apwg_trends_report_q1-q3_2015.pdf

[6] L. M. Form, K. L. Chiew, S. N. Sze, and W. K. Tiong, "Phishing email detection technique by using hybrid features," in *Proc. 9th Int. Conf. IT Asia (CITA)*, Aug. 2015, pp. 1–5.

[7] Microsoft. (2018). *Microsoft Security Intelligence Report*. [Online]. Available: https://clouddamcdnprodep.azureedge.net/gdc/gdcVAOQd7/original

[8] M. Hiransha, N. A. Unnithan, R. Vinayakumar, and K. Soman, "Deep learning based phishing e-mail detection," in *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal. (IWSPA)*, A. D. R. Verma, Ed. Tempe, AZ, USA, Mar. 2018.

[9] C. Coyotes, V. S. Mohan, J. Naveen, R. Vinayakumar, and K. P. Soman, "ARES: Automatic rogue email spotter," in *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal. (IWSPA)*, A. D. R. Verma, Ed. Tempe, AZ, USA, Mar. 2018.

[10] S. Sheng, B. Wardman, G. Warner, L. Cranor, J. Hong, and C. Zhang, "An empirical analysis of phishing blacklists," in *Proc. 6th Conf. Email Anti-Spam (CEAS)*, Sacramento, CA, USA, 2009, pp. 1–10.

[11] R. Verma and N. Hossain, "Semantic feature selection for text with application to phishing email detection," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Cham, Switzerland: Springer, 2013, pp. 455–468,

[12] G. Park and J. M. Taylor. (2015). "Using syntactic features for phishing detection." [Online]. Available: https://arxiv.org/abs/1506.00037

[13] R. Verma, N. Shashidhar, and N. Hossain, "Detecting phishing emails the natural language way," in *Proc. Eur. Symp. Res. Comput. Secur.* Berlin, Germany: Springer, 2012, pp. 824–841.

[14] A. Vazhayil, N. B. Harikrishnan, R. Vinayakumar, and K. P. Soman, "PED-ML: Phishing email detection using classical machine learning techniques," in *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal. (IWSPA)*, A. D. R. Verma, Ed. Tempe, AZ, USA, 2018, pp. 1–8.

[15] I. R. A. Hamid and J. Abawajy, "Hybrid feature selection for phishing email detection," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.* Berlin, Germany: Springer, 2011, pp. 266–275.

[16] A. Bergholz, J. De Beer, S. Glahn, M.-F. Moens, G. Paaß, and S. Strobel, "New filtering approaches for phishing email," *J. Comput. Secur.*, vol. 18, no. 1, pp. 7–35, 2010.

[17] J. Singh, "Detection of phishing e-mail," in *Proc. IJCST*, vol. 2, no. 1, 2011, pp. 547–549.

[18] A. Bergholz, J.-H. Chang, G. Paaß, F. Reichartz, and S. Strobel, "Improved phishing detection using model-based features.," in *Proc. CEAS*, 2008, pp. 1–11.

[19] X. Gu and H. Wang, "Online anomaly prediction for robust cluster systems," in *Proc. IEEE Int. Conf. Data Eng.*, Mar./Apr. 2009, pp. 1000–1011.

[20] C. N. Gutierrez *et al.*, "Learning from the ones that got away: Detecting new forms of phishing attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 6, pp. 988–1001, Dec. 2018.

[21] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 513–520.

[22] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proc. 1st Workshop Vector Space Modeling Natural Lang. Process.*, 2015, pp. 39–48.

[23] D. Bahdanau, K. Cho, and Y. Bengio. (2014). "Neural machine translation by jointly learning to align and translate." [Online]. Available: https://arxiv.org/abs/1409.0473

[24] T. Repke and R. Krestel, "Bringing back structure to free text email conversations with recurrent neural networks," in *Proc. Eur. Conf. Inf. Retr.* Cham, Switzerland: Springer, 2018, pp. 114–126.

[25] F. Chollet. (2016). *Keras*. [Online]. Available: https://keras.io/

[26] J. Zhang and X. Li, "Phishing detection method based on borderline-smote deep belief network," in *Proc. Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*. Cham, Switzerland: Springer, 2017, pp. 45–53.

[27] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, and F. A. González, "Classifying phishing URLs using recurrent neural networks," in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, Apr. 2017, pp. 1–8.

[28] 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.

[29] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Proc. AAAI*, vol. 333, 2015, pp. 2267–2273.

[30] *20Newsgroups*. Accessed: Dec. 23, 2018. [Online]. Available: http://qwone.com/~jason/20Newsgroups/20news-bydate.tar.gz

[31] *Fudan Set*. Accessed: Dec. 23, 2018. [Online]. Available: www.datatang.com/data/44139 and 43543

[32] *ACL Anthology Network*. Accessed: Dec. 23, 2018. [Online]. Available: old-site.clsp.jhu.edu/~sbergsma/Stylo/.

[33] *Stanford Sentiment Treebank*. Accessed: Dec. 23, 2018. [Online]. Available: nlp.stanford.edu/sentiment/

[34] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. AAAI*, 2017, pp. 2852–2858.

[35] M.-T. Luong, H. Pham, and C. D. Manning. (2015). "Effective approaches to attention-based neural machine translation." [Online]. Available: https://arxiv.org/abs/1508.04025

[36] M. F. Stollenga, J. Masci, F. Gomez, and J. Schmidhuber, "Deep networks with internal selective attention through feedback connections," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3545–3553.

[37] K. Xu *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2048–2057.

[38] K. Cho *et al.* (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation." [Online]. Available: https://arxiv.org/abs/1406.1078

[39] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2016, pp. 1480–1489.

[40] T. Mikolov, K. Chen, G. Corrado, and J. Dean. (2013). "Efficient estimation of word representations in vector space." [Online]. Available: https://arxiv.org/abs/1301.3781

[41] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[42] Y. Goldberg and O. Levy. (2014). "Word2vec explained: Deriving Mikolov *et al.*'s negative-sampling word-embedding method." [Online]. Available: https://arxiv.org/abs/1402.3722

[43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[44] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. OSDI*, vol. 16, 2016, pp. 265–283.

[45] *The First Security and Privacy Analytics Anti-Phishing Shared Task*. Accessed: Dec. 23, 2018. [Online]. Available: https://dasavisha.github.io/IWSPA-sharedtask/?tdsourcetag=s_pctim_aiomsg
[46] *Enron Email Dataset*. Accessed: Feb. 26, 2019. [Online]. Available: https://www.cs.cmu.edu/~enron/
[47] *Nazario's Phishing Corpora*. Accessed: Feb. 26, 2019. [Online]. Available: https://monkey.org/~jose/phishing/
[48] *The Dada Engine*. Accessed: Feb. 26, 2019. [Online]. Available: http://dev.null.org/dadaengine/
[49] A. El Aassal, L. Moraes, S. Baki, A. Das, and R. Verma, "Anti-phishing pilot at ACM IWSPA 2018," in *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal. (IWSPA)*, A. D. R. Verma, Ed. Tempe, AZ, USA, 2018, pp. 1–9.
[50] *Email—An Email and MIME Handling Package*. Accessed: Feb. 26, 2019. [Online]. Available: https://docs.python.org/3/library/email.html
[51] V. Ra, B. G. Hba, A. K. Ma, S. Kpa, and P. Poornachandran, "DeepAntiPhishNet: Applying deep neural networks for phishing email detection," in *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal. (IWSPA)*, A. D. R. Verma, Ed. Tempe, AZ, USA, 2018, pp. 1–11.

YONG FANG received the Ph.D. degree from Sichuan University, Chengdu, China, in 2010, where he is currently a Professor with the College of Cyber Security. His research interests include network security, Web security, the Internet of Things, big data, and artificial intelligence.

CHENG ZHANG received the B.Eng. degree in electronic information science and technology from Sichuan University, Chengdu, China, in 2017, where he is currently pursuing the M.A. degree with the College of Cybersecurity. His current research interests include Web development and network security.

CHENG HUANG received the Ph.D. degree from Sichuan University, Chengdu, China, in 2017, where he is currently an Assistant Research Professor with the College of Cyber Security. From 2014 to 2015, he was a Visiting Student with the School of Computer Science, University of California, CA, USA. His current research interests include Web security, span networks, social privacy, system security, and artificial intelligence.

LIANG LIU received the M.A. degree from Sichuan University, Chengdu, China, in 2010, where he is currently an Assistant Professor with the College of Cyber Security. His current research interests include malicious detection, network security, system security, and artificial intelligence.

YUE YANG received the B.Eng. degree in information security from Sichuan University, Chengdu, China, in 2018, where he is currently pursuing the M.A. degree with the College of Cybersecurity. His current research interests include Web development and network security.