Convolutional Neural Network Optimization for Phishing Email Classification

Cameron McGinley
School of Computing
Wichita State University
Wichita, Kansas, USA
cwmcginley@shockers.wichita.edu

Sergio A. Salinas Monroy
School of Computing
Wichita State University
Wichita, Kansas, USA
sergio.salinasmonroy@wichita.edu

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671531

***Abstract*—Phishing emails are one of the most common and effective tools that cybercriminals use to gain access to an organization's network or personal information. To detect these attacks, email service providers use a variety of tools and indicators, such as the URLs that attackers include in their email messages. However, cybercriminals are able to bypass these detection techniques by omitting URLs in their messages and instead engaging victims in a conversation to advance their attacks. In this paper, we investigate the performance of convolutional neural network (CNN) models that identify phishing attacks by analyzing only the text in the email messages. The models take as input an embedding of the text in the email's body and output a probability indicating the likelihood that the message is malicious. We evaluate several CNN architectures using real-world phishing emails and find that the best performing one can identify phishing attacks with an accuracy of 98.139%, recall of 98.125%, and precision of 98.269%.**

***Index Terms*—phishing, emails, deep learning, convolutional neural networks, word embedding, natural language processing**

I. Introduction

Cybercriminals can launch phishing attacks by sending emails with messages that entice potential victims to take actions that benefit the criminals. The email messages are sent to addresses obtained by the cybercriminals, either by compromising websites or buying them from other cybercriminals. Successful phishing attacks may persuade victims to send cryptocurrency to the criminal or reveal the login credentials to their email or bank accounts. To appear legitimate, cybercriminals will often craft their emails to impersonate well-known brands. F5 Labs' 2019 report on Phishing and Fraud indicates the most commonly impersonated brands are: Facebook, Apple, Chase, Office, WhatsApp, Paypal, Amazon, Microsoft, iCloud, and Office365 [1]. To improve the success rate of their phishing attacks, some cybercriminals use spear phishing attacks. A spear phishing attack is targeted toward a single individual. These targeted attacks may leverage personal information about the victim gathered from previous attacks or through open source intelligence.

The number of phishing attacks, and their financial damages, are constantly growing. According to the FBI's Internet Crime Complaint Center (IC3), there were 114,702 reported cases of phishing (more than any other type of cyber attack) during 2019 [2]. Likewise, the amount of stolen money through phishing attacks has also increased from $48.2 million in 2018 to $57.8 million in 2019, a roughly 20% increase. Due to underreporting, the real figures are likely much larger. Fig.1 shows the upward trend in both the number of phishing attacks and their financial damages. Therefore, detecting phishing emails and ultimately preventing them from reaching users' inboxes is critical to prevent cybercriminals from causing large financial losses and launching additional cyberattacks.

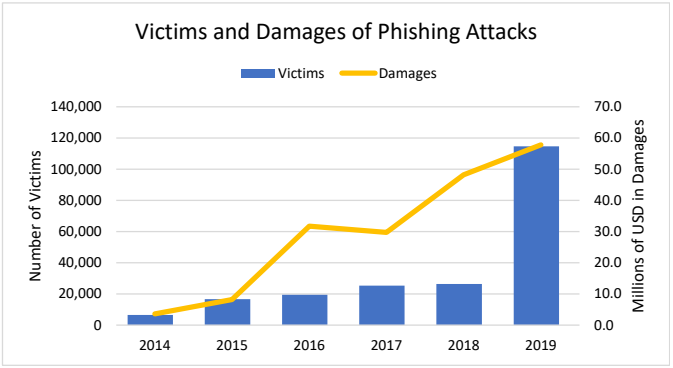

Fig. 1: Reported phishing attacks and their financial damages [2].

There have been important advances in the detection of phishing emails in the past few years. Akinyelu et al. and Yasin et al. use machine learning techniques, including random forests and support vector machines, to detect phishing emails with high accuracy [3], [4]. However, they require preprocessing of the email messages to extract certain features, which is time consuming due to the need for extensive testing to find the most effective features.

More recently, researchers have proposed deep learning. In deep learning, the email message is directly given as an input to a neural network, which then outputs the probability that the email message is malicious. Compared to machine learning, deep learning can find the relevant features of the email without the need for hand-crafted features [5], [6].

Unfortunately, cybercriminals can bypass existing deep learning detection techniques using relatively simple methods.

This material is based upon work supported by the National Science Foundation under Grant Numbers 1817537 (2018) and 1305059 (2013).

Specifically, previous phishing email detection methods rely on metadata such as links to phishing websites or attachments [7]–[9]. Thus, cybercriminals can bypass these methods by excluding these components from their phishing emails.

When phishing attacks are unable to use URLs, the attackers rely on the persuasiveness of their messages to incite users to perform an action on their behalf. Effective phishing messages evoke certain emotions in the victims, including urgency, appeals to authority, etc. To detect these types of phishing attacks, researchers have also used deep learning models that can analyze the semantic meaning of messages [10]–[13]. Although these techniques have successfully identified phishing emails, there are only a few studies that investigate the training complexity and efficiency of these deep learning models.

To address these gaps in the literature, we investigate the performance of convolutional neural network (CNN) architectures that detect phishing emails by only using the text in the email body, i.e., without relying on URLs or other information. Specifically, we implement CNNs with varying hyper-parameters including number of hidden layers, pooling layers, dropout values, output dimension, and training epochs and evaluate their accuracy, precision, and recall. We train our deep neural networks with three real-world data sets containing both legitimate and phishing emails. We find that the best performing architecture can achieve an accuracy of up to 98.139% using 0.2 dropout rate, 5 layers, 256 output dimension, and trained over 100 epochs. We observe that with only 5 layers the deep neural network still achieved a high accuracy of 98.147% but takes 61.3% less time to train than 10 layers which performed at 98.200% accuracy. Likewise, 100 epochs performed at an accuracy of 98.147% but uses 90.1% less time to train than 1000 epochs which performed at 98.213%.

II. Related Works

Due to the ever-growing threat of malicious emails [2], detection methods have received significant attention from researchers. Specifically, the research on phishing email detection can be broadly classified based on the type of machine learning techniques used and the email data that they analyze. The first efforts in phishing detection used traditional machine learning techniques such as random forests, support vector machines, and neural networks [3], [4]. These techniques require hand-crafted features that can then be analyzed by the machine learning methods, which is challenging and time consuming.

To overcome this challenge, deep neural networks, i.e., neural networks with multiple hidden layers, designed for text classification can be used to differentiate legitimate emails from phishing ones [10]–[14]. Compared to traditional machine learning, deep learning can analyze the raw email data without the need for researchers to design effective features, but it requires more computing resources to train and implement. Thus, it is important to find the deep learning architecture that offers the best trade-off between performance and training time.

A particularly successful type of deep learning model for text classification is the convolutional neural network (CNN) [15]–[18]. For example, Hassan et al. [15] classify sentences into separate categories, such as movie reviews as positive or negative using a CNN. They find that they can reduce the number of layers in the CNN to reduce the training computational complexity while trading only a small performance degradation. Hiransha et al. [14] demonstrate that accuracies of up to 98% can be achieved in text classification using CNNs. In this work, we use CNNs to detect phishing emails based on the text in their bodies.

A key component of deep neural networks for text classification is the embedding layer that translates the characters in the text to a numerical representation that can be used by the deep neural network. The most commonly used embedding is Word2Vec that assigns a vector of real numbers to each word in the input text. However, since phishing and legitimate emails may have spelling errors, some of their words may not have been assigned to a vector by the original Word2Vec implementation. To address this challenge, researchers have proposed to use character-level embedding. For example, Fang et al. propose a method of multilevel embedding using character-level and word-level embedding with Word2Vec [11], which is robust to spelling mistakes. Zhang et al. also uses character-level embedding and compares the performance against other popular methods [16].

Besides character level embedding, researchers have also proposed to use dropout layers to improve the robustness of the deep neural networks to spelling mistakes. This technique randomly drops units and their connections from the neural network during training to prevent overfitting [18]. Hinton et al. and Kim et al. also see improvements in convolutional neural networks while testing dropout rates, with relative performance improvements of 2% to 4% [17], [19]. The usage of dropout seems to be dependent on the task at hand and will require testing to perfect our own model.

There have been some efforts to identify the best performing deep learning architecture. Vinyakumar et al. [10] test and compare the accuracy of CNNs, recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and multilayer perceptrons (MLP). They find that their performance in terms of accuracy is comparable, i.e., within 2%. Coyotes et al. also test and compare CNN, RNN, and MLP [12], and they find that CNN compares similarly to RNN when headers are not used, whereas RNN performs better when headers are used. In both instances, CNN and RNN outperform MLP significantly.

Researchers have also compared the effectiveness of only employing the email header information to the effectiveness of using the body text. Nguyen et al. compares the exclusion and inclusion of the email header and finds an increase in accuracy from 98.1% to 99.0% when headers are included [13]. Hiransha et al. also tests this same question but finds the opposite results. Their results show an accuracy of 96.8%

without headers and 94.2% with headers [14]. Fang et al. also works with email headers and finds issues with correctly weighting the header and body [11]. The contradictory results may be explained by the varying abilities of different cybercriminals to craft messages that appear to come from a legitimate sender as well as the different data sets used in these works. Nonetheless, these works provide a performance baseline for different types of deep learning models in the area of phishing detection with message text.

III. Threat Model

In this section, we describe our assumptions about the attackers who send phishing emails. Specifically, we assume an attacker sends an email to a set of potential victims. The attacker aims to persuade the potential victims to perform an action that will result in a financial benefit for the attacker. The attacker can engage the victims in conversation by sending multiple emails. We assume the victims' email provider verifies the authenticity of the incoming emails with DKIM, SPF, and DMARC [20], [21]. Moreover, the victims' email provider also analyzes the body text for malicious URLs. We assume that the attacker is aware of the phishing and spam detection techniques employed by the victims' email service provider, and attempts to bypass them. To bypass these detection techniques the attacker avoids adding URLs to the body text and employs a legitimate email service provider that complies with SPF, DKIM, and DMARC to send its messages. Thus, to complete its attack, the attacker must draft persuasive email messages that convince the victim to perform some action on their behalf.

IV. Deep Neural Network Design

Due to the success of CNNs in text classification [15]–[18], we focus on this type of deep learning model in this work. In the following, we describe the CNN architectures that we consider to detect phishing emails launched by the attackers described in Section III. The models are all formed by a character-level embedding input layer, convolutional hidden layers coupled with pooling layers, and a sigmoid activation function output layer. We show the general architecture of the deep neural networks in Fig. 2.

In particular, a CNN for phishing detection works as follows. First, the characters of the body of an email message to be classified are converted to integers by the embedding layer. Our deep neural network architecture passes the email embedding to a set of convolutional layers each paired with a pooling layer. The convolutional layers extract features of the input, while the pooling layers calculate maximum values from the feature maps. The outputs of the pooling layers are flattened into a single vector to make up a fully connected layer. To make our phishing email detection robust against overfitting and less dependent on correct spelling, we apply a dropout layer to randomly drop a percent of the data. We then apply a dense layer that summarizes the results of the dropout layer into a scalar. The scalar represents the probability that the email is a phishing email. Finally, we use a sigmoid activation function to find the final classification decision.

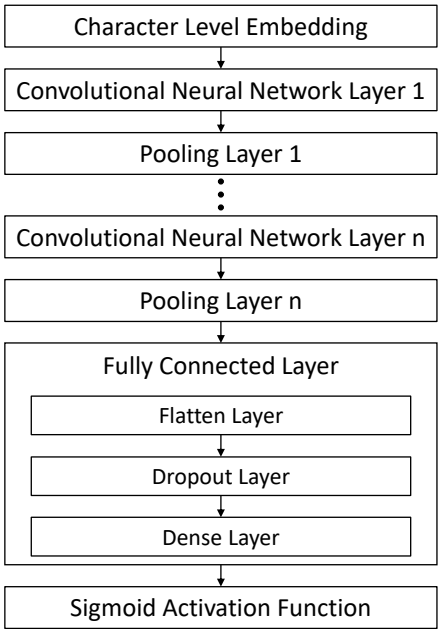

Fig. 2: Proposed deep neural network.

V. Training Datasets

We train the deep learning model described in Section IV with legitimate and phishing emails from three data sets: the Nazario phishing corpus [22], Enron-spam dataset [23], and the Enron email dataset [24]. The Nazario phishing corpus is a set of hand collected phishing emails that impersonate automated business or credit card companies. The Enron spam dataset is made of scam emails pulled from the Enron email dataset, many of which impersonate a personal contact. The legitimate emails from the Enron dataset are emails written by individuals and sent to others within the company. In total, we use 1,934 phishing emails and 1,870 legitimate emails.

We process the emails from the datasets with a Python script. We filter out the email header and subject data and only keep the body text. We choose to only use the body text due to previous research showing that it does not necessarily improve the performance of phishing email detection to use other features [11], [14]. Moreover, we remove abnormal inputs, such as HTML tags, extra spaces, and special characters, from the body text. All text is made lowercase and punctuation is ignored. Emails with fewer than 10 characters or with more than one thousand characters are removed. To avoid duplicates, we remove all forwarded messages.

VI. Experiment Results

In this section, we evaluate the detection performance of the deep neural networks with varying hyper-parameters described in Section IV in terms of accuracy, precision, and recall under various architecture settings.

A. Experiment Setup

To evaluate the performance of the deep neural networks, we train them under various settings with the datasets described in Section V. We vary the dropout rate of the dropout layers, the number of training epochs, number of convolutional/pooling layers, and the size of the output of the embedding layer. We use a batch size of 128 for all tests. We set aside 20% of the emails as testing data.

We measure the performance in terms of its accuracy, precision, and recall. Specifically, we denote the labeling of an email as a phishing email as a positive event (P), and the labeling of an email as legitimate as a negative event (N). We denote true positive, false positive, true negative, and false negative events as TP, FP, TN, FN, respectively. From these results, we study the accuracy, precision, and recall. Accuracy is used to measure the overall percent of correct classifications, positive and negative. Precision measures the percent of positive classifications (emails our model classified as malicious) that were classified correctly. Recall measures the percent of actual positive instances that were classified correctly. These measures are given by:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

We initialize the models with the following parameters: 5 layers, 0.2 dropout rate, 50 epochs, 128 output dimension. For each parameter that we evaluate, we train and test the deep neural network 10 times (30 if computation time allows and more iterations are needed to find a trend) and report the average accuracy.

B. Results

TABLE I: Dropout Rate Test Results

Dropout Rate	Accuracy	Recall	Precision	Time
0	.97608	**.97799**	.97500	0:04:50
0.1	.97819	.97738	.98037	0:04:51
0.15	.97805	.97713	.98042	0:04:52
0.2	**.97845**	.97778	.98075	0:04:52
0.25	.97477	.97454	.97639	0:04:51
0.35	.97490	.97685	.97338	0:04:52
0.5	.97582	.97348	**.98133**	0:04:51
0.7	.97166	.97294	.97158	0:04:52

5-layer CNN, 50 epochs, 128 output dim. Results averaged over 10 separate training/testing iterations.

We first evaluate the accuracy under varying values of dropout in Table I. We see that a dropout rate of 0.1 to 0.2 yields the highest accuracy for phishing detection. The accuracy decreases towards 0.5 when dropout rates increase beyond 0.2. Dropout rate is shown to have little to no impact on time, so there is no tradeoff to consider when exploring the rate. We fix the dropout rate to 0.2 for the remainder of our tests.

TABLE II: Epochs Test Results

Epochs	Accuracy	Recall	Precision	Time
50	.97845	.97778	.98075	0:04:52
100	.98147	.97767	**.98542**	0:09:32
500	.98042	.97974	.98208	0:47:57
1000	**.98213**	**.98351**	.98127	1:35:51

5-layer CNN, 0.2 dropout, 128 output dim. Results averaged over 10 separate training/testing iterations

Next, we show the effect of varying amounts of training epochs on the accuracy. Table II shows the accuracy under 50, 100, 500, and 1000 epochs. We observe that the accuracy increases with the number of epochs, reaching its maximum at 1000 epochs. We also see that at 100 epochs the accuracy is already 99.9% of the maximum at 1000 epochs. Computation time increases linearly, so 1000 epochs require ten times more computation time than 100 epochs. Due to the negligible performance gain above 100 epochs and the much larger computation effort needed, we identify 100 epochs as the optimal choice for our model.

TABLE III: Layers Test Results

Layers	Accuracy	Recall	Precision	Time
5	.98147	.97905	.98272	0:09:45
6	.98016	.98203	.98249	0:12:13
7	.98187	.97991	.98190	0:15:07
8	.98069	.98031	**.98377**	0:18:13
10	**.98200**	**.98212**	.98286	0:25:12
15	.97941	.97754	.98187	0:44:43
20	.98108	.98086	.98178	1:18:11

0.2 dropout, 100 epochs, 128 output dim. Results averaged over 30 separate training/testing iterations

We also test the accuracy of the deep neural network under increasing numbers of convolutional neural network layers and pooling layers. Table III shows the accuracy of the model under 5, 6, 7, 8, 10, 15, and 20 layers. The results in this table show that there is no discernable trend in accuracy based on the number of these layers. However, there is a positive correlation between the number of layers and computation time, so a lower number of layers is optimal in this regard. For these reasons, we use 5 layers in our model.

Lastly, we evaluate the accuracy under different output dimensions of the embedding layer and report them in Table IV. We test output dimensions 64, 128, 256, and 512. We see that an embedding dimension of 256 performs significantly better than 64 and 128 and slightly better than 512.

Overall, we observe that the deep learning architecture that achieves the highest accuracy has the following parameter values: dropout rate of 0.2, 100 epochs, 5 convolutional and

TABLE IV: Output Dimension Test Results

Output Dimension	Accuracy	Recall	Precision	Time
64	.97451	.97358	.97638	0:02:06
128	.97727	.97797	.97740	0:04:53
256	**.98014**	.97781	**.98219**	0:10:04
512	.97976	**.98000**	.98033	0:27:58

5-layer CNN, 0.2 dropout, 50 epochs. Results averaged over 30 separate training/testing iterations

TABLE V: Final Model Configuration

Accuracy	Recall	Precision
.98139	.98125	.98269

5-layer CNN, 0.2 dropout, 100 epochs, and 256 output dim. Results averaged over 30 separate training/testing iterations

pooling layers, and an embedding layer output dimension of 256. Its computed accuracy, recall, and precision are seen in Table V.

VII. Conclusions

In this paper, we have studied the problem of using deep learning to detect phishing emails purely using the body text. With additional features used by other techniques, cybercriminals can tailor their attacks to exclude these features or use them in a way to avoid detection on that specific feature. Our model aims for robust detection regardless of features beyond text. We find that the proposed CNN with character level embedding proved successful in the task of phishing email identification. The model reliably reached successful classification rates of over 98% on the data set of phishing and legitimate emails. The model performed well over a range of parameters, and the parameters with best results were used to create the final model. Specifically, the parameters that performed best were five convolutional and pooling layers, 0.2 dropout rate, 256 output dimension, and 100 epochs. When inaccurate, the model tended to hit equal amounts of false positives, or classification of an email as malicious when it is not, and false negatives. In future work, we plan to explore architectures that maximize the recall or the precision of the deep learning architecture depending on the priorities of the email service provider who analyzes the incoming email messages.

References

[1] D. Warburton and R. Pompon, "Phishing and fraud report," https://www.f5.com/labs/articles/threat-intelligence/2019-phishing-and-fraud-report, October 2019.

[2] Internet Crime Complaint Center (IC3), FBI, "2019 annual report," https://www.ic3.gov/media/annualreports.aspx.

[3] A. A. Akinyelu and A. O. Adewumi, "Classification of phishing email using random forest machine learning technique," *Journal of Applied Mathematics*, vol. 2014, 2014.

[4] A. Yasin and A. Abuhasan, "An intelligent classification model for phishing email detection," *arXiv preprint arXiv:1608.02196*, 2016.

[5] L. Deng and D. Yu, "Deep learning: Methods and applications," *Foundations and trends in signal processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[6] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35 365–35 381, 2018.

[7] S. Phomkeona and K. Okamura, "Zero-day malicious email investigation and detection using features with deep-learning approach," *Journal of Information Processing*, vol. 28, pp. 222–229, 2020.

[8] Q. Li, M. Cheng, J. Wang, and B. Sun, "LSTM based phishing detection for big email data," *IEEE Transactions on Big Data*, 2020.

[9] A. Bergholz, J. H. Chang, G. Paass, F. Reichartz, and S. Strobel, "Improved phishing detection using model-based features." in *CEAS*, 2008.

[10] V. Ra, B. G. HBa, A. K. Ma, S. KPa, P. Poornachandran, and A. Verma, "DeepAnti-PhishNet: Applying deep neural networks for phishing email detection," in *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal.(IWSPA)*. Tempe, AZ, USA, 2018, pp. 1–11.

[11] Y. Fang, C. Zhang, C. Huang, L. Liu, and Y. Yang, "Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism," *IEEE Access*, vol. 7, pp. 56 329–56 340, 2019.

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

[13] M. Nguyen, T. Nguyen, and T. H. Nguyen, "A deep learning model with hierarchical LSTMs and supervised attention for anti-phishing," *arXiv preprint arXiv:1805.01554*, 2018.

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

[15] A. Hassan and A. Mahmood, "Convolutional recurrent deep learning model for sentence classification," *IEEE Access*, vol. 6, pp. 13 949–13 957, 2018.

[16] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *Advances in neural information processing systems*, vol. 28, pp. 649–657, 2015.

[17] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[19] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[20] E. Derouet, "Fighting phishing and securing data with email authentication," *Computer Fraud & Security*, vol. 2016, no. 10, pp. 5–8, 2016.

[21] Z. Durumeric, D. Adrian, A. Mirian, J. Kasten, E. Bursztein, N. Lidzborski, K. Thomas, V. Eranti, M. Bailey, and J. A. Halderman, "Neither snow nor rain nor MITM... an empirical analysis of email delivery security," in *Proceedings of the 2015 Internet Measurement Conference*, 2015, pp. 27–39.

[22] J. Nazario, "The online phishing corpus," 2004.

[23] V. Metsis, I. Androutsopoulos, and G. Paliouras, "Spam filtering with naive Bayes - which naive Bayes?" in *CEAS*, vol. 17. Mountain View, CA, 2006, pp. 28–69.

[24] B. Klimt and Y. Yang, "The Enron corpus: A new dataset for email classification research," in *European Conference on Machine Learning*. Springer, 2004, pp. 217–226.