2018 IEEE International Conference on Data Mining Workshops (ICDMW)

Phishing Email Detection Using Robust NLP Techniques

Gal Egozi
Department of Computer Science
University of Houston
Houston TX, USA
geegozi@gmail.com

Rakesh Verma
Department of Computer Science
University of Houston
Houston TX, USA
rverma@uh.edu

***Abstract*—Even with many successful phishing email detectors, phishing emails still cost businesses and individuals millions of dollars per year. Most of these models seem to ignore features like word count, stopword count, and punctuations; they use features like n-grams and part of speech tagging. Previous phishing email research ignores or removes the stopwords, and features relating to punctuation only count as a minor part of the detector. Even with a strong unconventional focus on features like word counts, stopwords, punctuation, and uniqueness factors, an ensemble learning model based on a linear kernel SVM gave a true positive rate of 83% and a true negative rate of 96%. Moreover, these features are robustly detected even in noisy email data. It is much easier to detect our features than correct part-of-speech tags or named entities in emails.**

***Index Terms*—Phishing, Email, NLP**

I. Introduction

Phishing emails are a serious problem that still has no perfect solution. While there are good models today, no model is perfect. Since phishing emails can be very deceiving, it is vital to improve on the phishing detection models. These deceiving emails cause millions of dollars in damages; through the rise of file-encrypting ransomware, phishing attacks have become especially dangerous.

From the first infamous file-encrypting ransomware in 2013, Cryptolocker, these malware have become extremely popular. The basic idea of this type of ransomware is to encrypt user files and force them to pay a few hundred dollars. The idea is that the files are being held ransom, so the user will be more likely to pay. Some of these ransomware even damage the boot sector, which prohibits the user from even booting the device. File encrypting ransomware has become so popular that in 2016, the security company PhishMe found that over 90% of phishing emails contained it in both the first and third quarter of 2016 [2], [3].

To combat the phishing email menace, the authors propose models with new features. The authors show that the features are fairly successful; however, there will likely be greater success when these features are added into existing detectors. We focus on fundamental features of an email: stopwords, punctuation, word counts and uniqueness factors. It seems that in previous literature the fundamental features are ignored; however, they are not pointless.

This research was partly funded by NSF grants IIS 1659755, CNS 1319212 and DGE 1433817.

While there were many authors who devised phishing email detectors using NLP, few of them considered elements like punctuation and stopwords. The previous models that used such features only had them as a minor part of the detector. The proposed model, therefore, has a heavy focus on these types of features.

Unlike formal print media, such as news articles, papers, and books, emails are relatively unstructured. This partial lack of structure means that both part of speech recognition and named identity recognition struggle. Due to social media's complete lack of structure, part of speech recognition is much more difficult; however, named identity recognition can be much easier. The ease of named identity recognition comes from the simplicity and convenience of linking other users or pages from the same social network to a post. Sometimes, the social networks will add a link automatically to someone else's profile in certain situations. With the amount of people on common social networks, such as Twitter and Facebook, a named identity recognizer that just searches for links to other profiles from the same social network will likely perform very well. With email's combination of being partially unstructured and lacking the ability to link named identities, the tasks of part of speech tagging and named identity recognition are very difficult for a machine to perform.

II. Proposed Model

The proposed model consists of two phases: feature extraction and machine learning. The details for each portion of the detector are explained below.

A. Features

Because there is no current way to have a machine sort emails without extracting features, there must be a feature extraction component. This model currently uses 26 features to determine whether emails are phishing or ham. While some of the features may not seem effective, even features such as word count can be fairly effective on its own. Since four of the features use special terms that have not been used before, their definitions follow. After the definitions, the full list of features can be found.

1) Feature Definitions: Define the cumulative number of times a stem appears in all phishing emails (including repeats within the same email) to be p, with h being the same applied to the ham emails. Also define p_{unique} to be the number of times a stem appears in all phishing emails, excluding repeats within the same email, and h_{unique} to be the same for the ham emails. Also define tot to be the total number of times a stem appears in all emails, including repeats within the same email. tot_{unique} is defined to be the number of times a stem appears in all emails without repeats from the same email. None of these quantities have any normalization applied on them; they are all raw values.

Definition II.1. Difference Measure of a stem
$\max(p - h, 0)$

Definition II.2. Difference Measure of an email

$$\sum_{e \in \text{stems in the email}} (\text{The difference measure of } e)$$

Definition II.3. Unique Difference Measure of a stem $\max(p_{\text{unique}} - h_{\text{unique}}, 0)$

Definition II.4. Unique Difference Measure of an email

$$\sum_{e \in \text{unique stems in the email}} (\text{The unique difference measure of } e)$$

Definition II.5. Phishing Ratio of a stem

$$\begin{cases} \frac{p}{tot} & tot > 0 \\ 0 & tot \leq 0 \end{cases}$$

Definition II.6. Phishing ratio of an email

$$\sum_{e \in \text{stems in the email}} (\text{The phishing ratio of } e)$$

Definition II.7. Unique Phishing Ratio of a stem

$$\begin{cases} \frac{p_{unique}}{tot_{unique}} & tot > 0 \\ \\ 0 & tot \leq 0 \end{cases}$$

Definition II.8. Unique Phishing Ratio of an email

$$\sum_{e \in \text{unique stems in the email}} (\text{The unique phishing ratio of } e)$$

2) List of Features: The first four features are used to predict the likelihood a certain stem will appear in phishing and ham emails. They compare an email to simple models of both phishing and ham emails to determine the appropriate category for an email. While the non-unique versions of these features contain more information about the emails, the unique versions try to prevent common words in all emails from clogging the models. In all four of these features, stopwords are kept to retain more information about the emails. If any of these four features are high, then the email is likely phishing. Since none of these features have a maximum, machine learning must be used to determine a suitable boundary for phishing email detection. Unlike the remaining features, these four features are the closest features in the model to the features most commonly used, since they compare word frequency in phishing emails versus ham emails.

In some of the features, there is a comparison between the count of a feature and the unique count of the feature. This paper uses uniqueness like a unary operator, that can be applied to a feature. When the unique 'operator' is applied, then the referenced feature analyzes the unique component of the feature. In this paper, the uniqueness operator is called the uniqueness factor. The definitions for the following four features are given above.

1) Difference Measure: The difference measure is used to determine how likely an email is a phishing email by comparing the number of times a stem appears in a phishing email to the number of times the stem appears in a ham email.
2) Phishing Ratio: The phishing ratio uses the ratio between phishing and total appearances of a stem to help determine the likelihood of an email being phishing.
3) Unique Difference Measure: The unique difference measure is similar to the difference measure, but it does not consider multiple appearances of a stem in an email.
4) Unique Phishing Ratio: While the unique phishing ratio is similar to the phishing ratio, it does not consider multiple appearances of the same stem in an email.

Unlike the previous features, these features are novel. These features are word, stopword, and punctuation counts as well as unique versus non-unique variants of these quantities and ratios between them.

5) Word Count: The word count is the total number of words in the email. While there is no maximum, the word count will always be an integer greater than or equal to zero.
6) Stopword Count: The stopword count is the number of stopwords in an email. Stopwords are the words that contribute the least meaning to an email. For this paper, the stopword list used is [4]. The number of stopwords will be at least zero, and at most the number of words in the email.
7) $\frac{\text{Stopword Count}}{\text{Word Count}}$: This ratio measures the relative frequency of stopwords in an email. This ratio, which is bounded from zero to one, will have higher magnitudes when larger portions of the email are stopwords. An email with ratio one contains purely stopwords, while an email with ratio zero contains no stopwords. Unless the word count is extremely low, when this ratio gets very close to either zero or one, then the email is likely phishing.
8) Unique Stopword Count: The unique stopword count measures the number of times a stopword appears in an email. While the theoretical maximum of this number is 571, emails are not long enough to use all the stopwords. Even the longest email in the dataset would need over

one unique stopword every four words in the email. Therefore, unless an email is extremely long, a unique stopword count that is close to 571 may be suspicious.

9) $\frac{\text{Unique Stopword Count}}{\text{Stopword Count}}$: This ratio quantifies the uniqueness of the stopwords in an email. In emails with many repeated stopwords, this ratio will be low; however, when there are many unique stopwords, the ratio will be high. This ratio has no correlation to the word count of an email. As most of the stopwords are fairly common words, the separation of suspicious regions is not as clear. However, the ratio should not be extremely close to zero or extremely close to one for ham emails, as a ratio extremely close to one indicates little to no repeated stopwords, while a ratio extremely close to zero indicates that nearly all the stopwords are the same.
10) $\frac{\text{Unique Stopword Count}}{\text{Word Count}}$: This ratio measures the number of unique stopwords per word in the email. With many unique stopwords, this ratio is high; however, when there are few stopwords, this ratio can get close to zero.
11) Punctuation Count: The punctuation count measures the number of punctuation symbols in an email. While it may be difficult to guess the patterns by hand, there are likely patterns that separate phishing emails from ham emails. Apostrophes inside of words and other similar uses of punctuation do not count as punctuation.
12) $\frac{\text{Punctuation Count}}{\text{Word Count}}$: This ratio measures the average number of punctuation per word. Most sentences will have between one to three punctuations, which sets appropriate limits for punctuation. If this ratio is too low or too high, this may indicate an informal email (such as to a friend or close family) or a phishing email. However, some people write emails with many punctuation marks, as an example, “Look at my paper!!!!!”. The algorithms used need to compensate for differences in writers. If there are many email communications between two people, this can be a feature used to determine if an email is phishing when the two people communicated with each other many times before. While this ratio is not bounded, a ratio greater than one or a ratio smaller than $\frac{1}{100}$ may indicate a suspicious email.
13) $\frac{\text{Punctuation Count}}{\text{Stopword Count}}$: This measures the average number of punctuation marks per stopword. This ratio should be larger than the ratio of $\frac{\text{Punctuation Count}}{\text{Word Count}}$, or otherwise the email is likely suspicious. While this ratio is not bounded, a ratio of lower than $\frac{1}{100}$ or a ratio higher than 5 may be suspicious. Like the ratio of $\frac{\text{Punctuation Count}}{\text{Word Count}}$, there can also be variance due to the author writing the emails.
14) $\frac{\text{Punctuation Count}}{\text{Unique Stopword Count}}$: This ratio measures the average number of punctuation marks per unique stopword. This is one of the more unique features, and it is difficult to tell boundaries for suspicion. The range of this function is all non-negative rational numbers with denominator less than or equal to 571.
15) Unique Punctuation Count: The unique punctuation count measures the unique number of punctuation symbols in an email. While it is unlikely for an email to use all 32 punctuation symbols, it is possible. A ham email with all 32 symbols should have at least 32 words in the email; however, there may be ham emails where this is not so.
16) $\frac{\text{Unique Punctuation Count}}{\text{Word Count}}$: This ratio shows the average number of different punctuation marks per word in the email. Since there are only 32 punctuation types, an email must be short if this ratio is close to or above one.
17) $\frac{\text{Unique Punctuation Count}}{\text{Stopword Count}}$: This ratio shows the average number of unique punctuation marks per stopword. This is one of the more interesting features, since there is not a direct or clear correlation between this ratio and the number of words in an email.
18) $\frac{\text{Unique Punctuation Count}}{\text{Punctuation Count}}$: This ratio is a measurement of repeated punctuation marks. Therefore, a low ratio indicates many repeated punctuation marks, while a high ratio indicates that more punctuation types were used with fewer repeats. A low ratio may occur in informal emails, such as “Look at my paper!!!!!”, long and formal emails (most likely heavy usage of the period character), or phishing emails.
19) $\frac{\text{Unique Punctuation Count}}{\text{Unique Stopword Count}}$: This measures the average number of unique punctuation marks for every unique stopword. As a new feature in phishing email detection, this ratio has interesting properties. Because of the complex and intricate properties of this ratio, it is difficult to determine bounds for this ratio.
20) Unique Word Count: The unique word count is the number of distinct words in an email. While the unique word count may not be an effective feature on its own, it can lead to interesting ratios. It is therefore kept as more of a baseline feature than anything else.
21) $\frac{\text{Unique Word Count}}{\text{Word Count}}$: This ratio measures how unique the words are in an email. This is one of the more intricate features in the model, as a ham email should have many unique words with some of them repeated, especially stopwords. If too few of the words are repeated, then the email is likely a phishing message (as in the ratio would be too close to zero); however, if the email contains too many repeated words, then the email is likely spam (as in the ratio would be too close to one).
22) $\frac{\text{Stopword Count}}{\text{Unique Word Count}}$: This ratio measures the average number of stopwords for every unique word. When this ratio is low, there are many more unique words than total stopwords; however, when this ratio is high, there are many more stopwords than unique words. If this ratio is high on a particular email, this means that there are many repeated stopwords, and is likely a spam or phishing message.
23) $\frac{\text{Unique Stopword Count}}{\text{Unique Word Count}}$: This ratio measures the average number of unique stopwords for every unique word. If this ratio is high, then there are many different stopwords and few unique words. As the number of unique words in an email increase past 571 words, this ratio becomes limited

as the number of unique stopwords can only reach 571. If the unique stopword count is low, but the ratio is high, then the email is likely spam. Since this ratio deals with uniqueness, its properties are more complex.

24) $\frac{\text{Punctuation Count}}{\text{Unique Word Count}}$: This measure shows the average number of punctuation marks per unique word. The graph of this feature versus word count would likely be very noisy for low value of words; however, for larger word counts the value of this ratio would start to increase.
25) $\frac{\text{Unique Punctuation Count}}{\text{Unique Word Count}}$: This feature is a measure of the ratio of the average number of unique punctuation marks per unique word. Because both measures are unique, it is difficult to predict any patterns for this ratio in an email, as this ratio may be significantly different for two similar emails.
26) Unique Stem Count: The unique stem count is a measure of the number of stems in an email with no repeats.

B. Machine Learning Models

Since different machine learning models function best in specific sets of data, 17 models were tested. Out of the 17 tested, 14 gave acceptable results. Below is a definition of the weighting scheme used throughout the models and results. *All models and metrics will be explicitly labeled as weighted or unweighted, and this is the only weighting scheme used in the entire paper.*

Definition II.9 (Weighted)**.** A weighted model or metric indicates that the emails were weighted. The scheme used to weigh the emails is shown below.

$$W_{\text{phish}} = \frac{\text{Total emails}}{\text{Phishing Emails}}$$

$$W_{\text{ham}} = \frac{\text{Total emails}}{\text{Ham Emails}}$$

In the above equations, W_{phish} is the weight of an individual phishing email, and W_{ham} is the weight of an individual ham email.

Remark. This weighting scheme is used when training machine learning models as well as for the evaluation metrics determining the accuracy of the models.

Since there are many more ham emails than phishing emails, this weighting scheme was chosen to ensure that the set of ham emails and the set of phishing emails are equally weighted. Even in cases where the dataset was weighted almost equally, or there would be more phishing emails than ham emails, this weighing scheme would still ensure that the set of all phishing emails and the set of all ham emails have an equal weight. For the weighted models, the weighing scheme was applied to the training data to prevent the model from claiming everything as ham emails. Here are the models used:

1) Unweighted Decision Tree
2) Weighted Decision Tree
3) Unweighted Multinomial Naive Bayes
4) Weighted Multinomial Naive Bayes
5) Unweighted Logistic Regression
6) Weighted Logistic Regression
7) Unweighted Neural Network
8) Unweighted SVM (RBF kernel) *Failed model*
9) Weighted SVM (RBF kernel) *Failed model*
10) Unweighted SVM (Linear kernel)
11) Weighted SVM (Linear kernel)
12) Unweighted SVM (Polynomial kernel)
13) Unweighted SVM (Sigmoid kernel) *Failed model*
14) Unweighted Gaussian Naive Bayes
15) Weighted Gaussian Naive Bayes
16) Unweighted Bernoulli Naive Bayes *Failed during Ensemble Learning*
17) Weighted Bernoulli Naive Bayes *Failed during Ensemble Learning*

The models which have the *Failed model* indication are models that gave a 90% or higher false positive rate and were therefore not used during any further evaluation. The models labeled with *Failed during Ensemble Learning* indicate models that gave either a 0% true positive rate or a 0% true negative rate. There could be several reasons for this divergence (e.g., the class imbalance or suboptimal hyperparameter values), we leave the determination of the causes of this phenomenon to future work. The two models that diverged during ensemble learning performed fairly well during the single-model phase. These two Bernoulli Naive Bayes models could have failed because of potential underflow issue of the computations of probabilities of an email being in a certain class. These probabilities were determined by the models from the single-model phase.

For the ensemble learning, the probability of an email being either ham or phishing was combined to form a set of 28 features. These set of 28 features were then used in models with the same types as the 14 successful models.

III. Results, Analysis, and Discussion

A. Dataset

The dataset used was the one used for the IWSPA competition [5]. This dataset contains both a training and testing dataset. The training dataset contains 3865 ham emails and 735 phishing emails, while the testing dataset contains 3824 ham emails and 475 phishing emails. The ham emails came from various Wikileaks sources as well as SpamAssassin, while the phishing emails came from the Nazario phishing corpora, the IT websites of various universities, and some generated phishing emails using the Dada engine.

B. Results

Throughout the entire results section, shorthand notation is used. The notation is described below.

bnb	Bernoulli Naive Bayes
dt	Decision Tree
f	f_1 Score
gnb	Gaussian Naive Bayes
lr	Logistic Regression
lsvm	Linear kernel SVM

mt	Metric (Which one of the six metrics: TPR, TNR, precision, recall, f_1 score, or area under curve)
pr	Precision
rc	Recall
rs	Run set (The set the model was run on)
te	Testing Dataset
tnr	True Negative Rate (Negatives are ham or non-phishing emails)
tpr	True Positive Rate (Positives are phishing emails)
tr	Training Dataset
uw	Unweighted
w	Weighted

Table I shows selected results for 5 models after running a single machine learning classifier for each of the 17 model types used. Based on Table I, it seems that the models can be divided into four categories: a failed model, a model that can successfully identify phishing emails, a model that can successfully identify ham emails, and a model that is relatively equally strong in both. From the five selected models, the unweighted Gaussian Naive Bayes is the worst, as it only detects 59% of phishing emails; therefore, such a model would only be good for people who are extremely cautious. With its 99% phishing detection rate, the weighted decision tree could be used in businesses with workers who are less aware about phishing emails. With only a 77% true negative rate, it should not be used in critical systems, such as in the US presidential office. In general, Table I shows that the features chosen were relatively good, as they worked well on most models. Since the models have different strengths and weaknesses, ensemble learning seemed like a viable idea to fix some of the problems with the models.

To generate the ensemble learning models, a 28 feature vector was generated for each email. These features are the probability of an email being phishing or ham, as determined by each of the 14 successful models. For each of the 14 successful models, an alternate model of the same type was made. These alternate models were then trained on the whole training dataset, using the 28-feature vector as its input.

Table II shows the results for four of the 14 models. These models were created when the probabilities of an email being a certain class were put into 14 new machine learning classifiers. Since each of the initial models generate 2 features per email, these new models have 28 features per email. Based on these results, it seems that the Weighted Linear SVM is the best model; however, this may be due to over training. Unlike the machine learning models used without ensemble learning, the ensemble learning models favored the ham emails (negatives). This table supports the conclusion that the features are effective in detecting phishing emails.

C. Feature Significance Evaluation

With any model containing new and novel features, it is vital to determine the importance of these features. Figure 1 shows the results of running Mutual Information on both the training dataset, the test dataset, and the entire dataset (the training and testing combined). The datasets used come from [5], and the numbering for the features is the same as above. While the most significant features (the first four) are the more traditional features (based on previous work), the new features are fairly important. The importance is statistically significant.

TABLE I
THIS TABLE SHOWS SOME OF THE RESULTS FROM RUNNING A SINGLE MACHINE LEARNING CLASSIFIER ON THE DATA. ALL NUMBERS IN TABLE ARE PERCENTAGES.

mt	rs	w dt	uw bnb	w lsvm	w bnb	uw gnb
tpr	uw te	99	85	97	85	59
tnr	uw te	77	83	83	83	90
pr	uw tr	100	55	97	55	49
pr	uw te	87	39	78	39	21
pr	w tr	100	87	99	87	84
pr	w te	98	84	97	84	69
rc	uw tr	100	96	99	96	98
rc	uw te	77	83	83	83	90
rc	w tr	100	96	99	96	98
rc	w te	77	83	83	83	90
f	uw tr	100	70	98	70	66
f	uw te	81	53	80	53	34
f	w tr	100	91	99	91	90
f	w te	86	84	89	84	78
auc	uw tr	100	91	99	91	89
auc	uw te	88	84	90	84	75
auc	w tr	100	91	99	91	89
auc	w te	88	84	90	84	75

TABLE II
RESULTS FROM RUNNING THE ENSEMBLE LEARNING CLASSIFIERS. ALL NUMBERS IN TABLE ARE PERCENTAGES.

mt	rs	w lsvm	w lr	uw gnb	w gnb
tpr	uw te	83	82	81	82
tnr	uw te	96	97	97	97
pr	uw tr	97	97	97	97
pr	uw te	72	75	77	74
pr	w tr	99	99	99	99
pr	w te	96	96	97	96
rc	uw tr	99	99	99	99
rc	uw te	83	82	81	82
rc	w tr	99	99	99	99
rc	w te	83	82	81	82
f	uw tr	98	98	98	98
f	uw te	77	78	79	78
f	w tr	99	99	99	99
f	w te	89	89	88	89
auc	uw tr	99	99	99	99
auc	uw te	89	90	89	89
auc	w tr	99	99	99	99
auc	w te	89	90	89	89

IV. SECURITY ANALYSIS

Since phishers constantly try to avoid detection, they may attempt to reverse-engineer the proposed method. Their challenge will be to not only fool the human, but also the detector. Through careful manipulation of the wording in an email, the phisher may be able to fool the detector; however, a phishing email that does not fool humans is ineffective. Since the phisher does not know the exact parameters of the model, the phisher must guess to find the optimal model. By using a

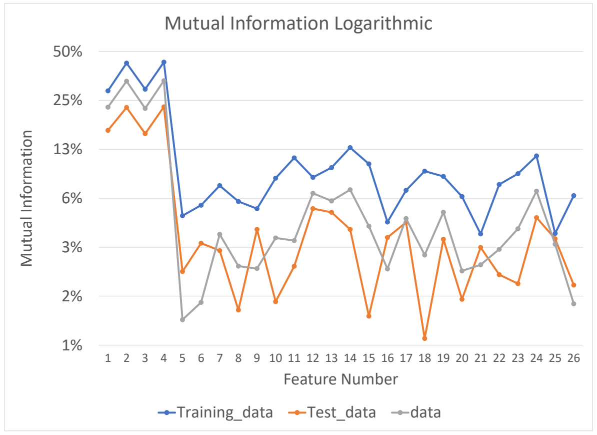

Fig. 1. The graph of Mutual Information. The scale on the y-axis is logarithmic.

blacklist, a repeat attack can be prevented. An email address, IP address, and other such data would be recorded in the blacklist to attempt to prevent the attacker from repeating the same attack. Since a detector can detect false positives, the detector would remove the record if a user marks the email as ham. With the attempted brute forcing, it is likely that the incoming email server will detect and stop the behavior; therefore, most of the emails can be caught. By the time the phisher would be able to pass the system, the resulting email would not be very effective, especially since it is unlikely the email would ever reach the user's inbox.

V. Related Work

While there has been some work done in phishing email detection using NLP [5]–[7], there is little work that considers the stopwords themselves. One example is [6], which removes the stopwords once the email has been tagged. That paper builds a model with lexical analysis, named identity recognition, and part of speech tagging. This paper also analyzes the way a word is used. Even with all of these advanced textual features, it may seem that textual analysis could be redundant. The accuracy of the header and link analysis are nearly identical to each other and to the accuracy of the entire model for all tested datasets. In [7], the authors identify phrases that distinguish phishing and legitimate emails. As far as punctuation, there are NLP techniques which analyze punctuation; however, most papers, such as [1], only have punctuation as a minor component of the detector. It uses many complex features, such as readability index and part of speech tagging. While it may work as a phishing email detector if the two parties have a long history of interactions, it would be very difficult to implement. The noisy nature of emails means that better models for email analysis would need to be developed before using this method. These analysis methods include part of speech taggers and named identity recognition methods. Since most other models depend on other features, the proposed model is therefore unique.

VI. Conclusion

We created phishing email detectors using novel features. The detector was able to properly identify over 80% of phishing emails and 95% of ham emails, while only using 26 features. With a significant portion of the features focusing on word counts, stopword counts, punctuation counts, and uniqueness, most of the features are therefore novel.

VII. Future Work

While the model is fairly successful, it can be improved. With the small number of features, the model does fairly well at classifying improvements; however, a better model can be made by including more features and more emails in the dataset or by using more datasets. Some additional features include evaluating some of the models without stopwords, adding features pertaining to n-grams and skip n-grams with $n > 1$, and evaluating the appearance of a particular word. Since the current model takes a few hours to run, the current model would need significant optimizations before adding more features or more emails. While some feature evaluation and selection is included, more would be better. This is especially important if more features were to be added. Without this step, the contributions of each feature cannot be determined. If features that contribute little significance are removed, the model will likely run faster. One final area of further work is in hyperparameter tuning. Since some of the models were not able to produce successful results, hyperparameter tuning could help improve those models.

Acknowledgments

The authors would like to thank Ayman El Aassal, Daniel Lee and members of the ReDAS Laboratory for helping with this research.

References

[1] David Guthrie, Louise Guthrie, Ben Allison, and Yorick Wilks. Unsupervised anomaly detection. In *IJCAI*, pages 1624–1628, 2007.
[2] PhishMe. Q1 2016 sees 93% of phishing emails contain ransomware. https://cofense.com/q1-2016-sees-93-phishing-emails-contain-ransomware/, 2016.
[3] PhishMe. Ransomware delivered by 97% of phishing emails by end of q3 2016 supporting booming cybercrime industry. https://cofense.com/ransomware-delivered-97-phishing-emails-end-q3-2016-supporting-booming-cybercrime-industry/, 2016.
[4] Gerard Salton and Chris Buckley. Stopword list 2. http://www.lextek.com/manuals/onix/stopwords2.html.
[5] Rakesh Verma and Avisha Das, editors. *Proceedings of the 1st Anti-Phishing Shared Task at 4th ACM IWSPA (IWSPA-AP)*, number 2124 in CEUR Workshop Proceedings, Aachen, 2018.
[6] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. Detecting phishing emails the natural language way. In *European Symposium on Research in Computer Security*, pages 824–841. Springer, 2012.
[7] Rakesh M. Verma and Nabil Hossain. Semantic feature selection for text with application to phishing email detection. In *Proc. 16th International Conference on Information Security and Cryptology ICISC, Revised Selected Papers*, pages 455–468. Springer, 2013.