Phishing Detection System

Version 1.0

Shaefer Drew

Abstract

The problem being addressed involves a common threat in cyber security: phishing. Phishing is defined as ”Communications with a human who pretends to be a reputable entity or person in order to induce the revelation of personal information or to obtain private assets” (Chio and Freeman, 2018). As phishing attacks become more prevalent, more advanced detection and filtering systems are being demanded by companies and email providers. My goal is to use natural language processing to detect phishing in emails, classifying each email as a phishing attempt or not in order to filter phishing attacks and prevent them from reaching the end user. Here I show an effective approach to classifying phishing emails. Using Netcraft link verification combined with Support Vector Machine (SVM) classification, this system outperforms other researched phishing email detection methods as well as popular text classification methods.

1 Introduction

Spam itself makes up over 50% of all emails sent worldwide (spa, 2018)[1]. Phishing is one of the most common types of spam email. Phishing is where someone poses as a legitimate figure in order to provoke someone to reveal sensitive information. Employees get dozens of emails each month that are considered phishing attempts. An example of a phishing email would be someone posing as a representative of a well-known bank, under a fake email address, asking you to verify or change your online banking login credentials (for the sake of stealing them). Phishing approaches have become more sophisticated and less detectable throughout the years. Additionally, with the digitization of commerce throughout the world, businesses are more exposed than ever to phishing attempts.

Today, phishing prevention takes many forms. The most common form is spam filtering. This can be effective; however, fails to filter out the more sophisticated phishing attempts. In using social engineering, phishing emails can often appear to be legitimate emails and remain undetected by spam filters. Another popular technique many companies employ to combat phishing is cyber security education. While educating people to recognize a phishing email can be beneficial, people can still be fooled.

Instead of leaving classification in the hands of employees, I hope to filter out phishing emails before they reach their targets by using natural language processing. My system reads through emails, combining link verification and machine learning classification to label each email as a phishing attempt or not. The system can read through large email datasets of multiple formats. It begins by extracting the URLs embedded in each email and verifying them as legitimate. It does so using the Netcraft link verification toolbar. This phishing extension analyzes the legitimacy of a website using multiple measures and returns a risk score. The system then combines this step with word vectorization of the subject and body of the email, TF-IDF transformation, and ultimately uses a Support Vector Machine Classifier to label phishing emails.

My approach not only considers and tests multiple word classification methods which are proven to be effective, but it also verifies links as a second firewall in case the email text itself fools the classifier. In using this 2 step approach, I was able to outperform other phishing detection systems, such as Netcraft (by itself) and SEAHound in terms of precision and recall. Many corporations, email servers, and cyber security firms could use this to

[1] `https://www.statista.com/statistics/420391/spam-email-traffic-share/`

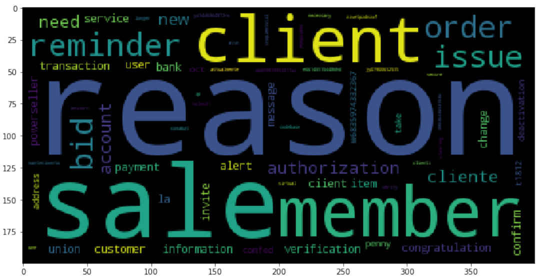


Figure 1: Phishing Email Word Cloud

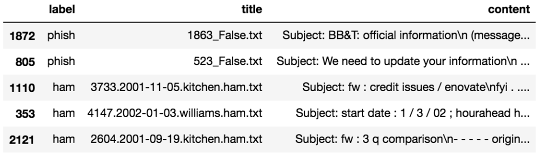
their benefit.

2 Problem Definition and Data

Nearly every email host has a system which sorts emails into different categories. One of the functions of each system is filtering out spam email. While spam is a problem in and of itself, with over half of email traffic being spam, I hope to narrow my focus on a specific type of spam, being phishing attempts. Phishing attempts are often much harder to recognize than spam as a whole because the hacker tends to specialize phishing emails for certain organizations. Corporations are targeted frequently with these types of attempts. My system will categorize emails as phishing or not in order to reduce vulnerability.

I am concatenating data sources from the Online Phishing Corpus and Enron email datasets to form a large dataset of emails labeled ”ham” and ”phish”. The Enron dataset was originally contained in text files whereas the Online Phishing Corpus dataset was contained in mbox files. The ham data is the data from Enron, which I pull from the ham folder. The phish data is the data from the phishing corpus, which contains a set of phishing emails. My system extracts the URLs from the mbox files and converts individual emails to text file, so as to be consistent with the Enron dataset.

The combined dataset contains 10,000 emails, half of which are ham and half of which are phishing emails. Each email is in text format and includes the subject, body, and date. The dataset includes the ham or phish label and a text blob of the email. When working through the data, I explore different methods of extraction and tokenization. The Enron dataset is observably cleaner than the Phishing Corpus dataset because not all mbox emails were presented in the same exact format, making it difficult to cleanly extract only email body and header text from each email.



	label	title	content
1872	phish	1863_False.txt	Subject: BB&T: official information\n (message...
805	phish	523_False.txt	Subject: We need to update your information\n ...
1110	ham	3733.2001-11-05.kitchen.ham.txt	Subject: fw : credit issues / enovate\nfyi . ....
353	ham	4147.2002-01-03.williams.ham.txt	Subject: start date : 1 / 3 / 02 ; hourahead h...
2121	ham	2604.2001-09-19.kitchen.ham.txt	Subject: fw : 3 q comparison\n- - - - - origin...

Figure 2: Combined Dataset

3 Related Work

The first paper I found uses an approach that expands on the Netcraft approach of URL verification. Their system, SEAHound, performs the same link analysis and, in addition, performs semantic analysis of verb-direct object pairs in sentences to classify them as bad, urgent, or generic (Peng et al., 2018). I use the same data sources for ham and phishing emails. I test different approaches and combinations of approaches to classification and verification. My ultimate system uses a similar link analysis approach with Netcraft and uses SVM classification on the remaining predictions as opposed to predicting the tonality of each sentence.

Another paper focuses on spam classification as opposed to phishing classification. It uses different data sources; however, it also uses a Naive Bayes approach in addition to N-gram modeling (Giyanani and Giyanani, 2014). I explore different types of topic modeling and classification techniques.

The 3rd paper I read uses one of the datasets I am looking into, the Enron dataset. Its goal is to correctly classify ham vs spam emails. It tests numerous techniques, including SVM and TF-IDF similarity in addition to Multinomial Naive Bayes (Klimt and Yang). I test both of these approaches in the classification stage of my filtering system.

Another source that's been very helpful in formulating my aproach to the email classification problem is the book *Machine Learning and Security* (Chio and Freeman, 2018). It works with the Trec Spam Dataset and experiments with different classification methods to predict spam. I took some of their approaches into consideration, especially their multinomial naive bayes approach.

4 Methodology

The first step of building the system was organizing and reading email data. Using the Phishing Corpus data as phishing data and the Enron data as non-phishing, I was able to generate a script which

Data	Source
The Online Phishing Corpus Phishing Emails	https://monkey.org/~jose/phishing/
Enron Email Dataset	http://www2.aueb.gr/users/ion/data/enron-spam/

Table 1: Examples of NLP tasks that you could choose to work on for your project. The last three tasks are very recent tasks that have details and data on the website but no papers yet. You can work on something cutting edge!

read through both formats.

There were many challenges in extracting purely the body and headers of the Phishing Corpus emails, due to their inconsistent format. I expanded on code from the SEAHound method in order to do so (Peng et al., 2018) [2].

The first checkpoint of my system revolved around link analysis. For this, I turned towards the Netcraft anti-phishing extension, which has shown to be effective in identifying phishing websites relative to other anti-phishing toolbars (Cranor et al., 2006). Netcraft offers many services in its link analysis process, including a database of known and reported phishing websites, malware scans, domain and host information, and a final risk score which considers all the information at hand [3]. In each of the email formats (text and mbox), I developed a separate scraping script which used regex to extract URLs found in each email.

In order to properly connect to the anti-phishing toolbar and automate the link analysis and verification process, I was able to use Selenium Webdriver to collect HTML data and retrieve risk scores. If a site had a risk score of 2 or above, I labeled the email as phishing and re-named the file with a binary value of true at the end. When reading the email data into a dataframe, a new column was constructed to signify whether the email contained a malicious link or not. This was used to label an email as a phishing email and took precedent over the classification algorithm's results. If the algorithm labeled an email as ”ham” which was ”phish” according to Netcraft, the final predicted label would be ”phish”. In this dataset, the Netcraft step was able to identify 322 out of the 5000 phishing emails.

When reading in the email data, I had 2 separate scripts for converting ”ham” data and ”phish” data into dataframes. These involved the title of the file, its label (ham vs phish), and the content of the email (header and body). I concatenated the 2 dataframes into a larger dataframe, consisting of 5000 phishing emails and 5000 non-phishing emails. For the phishing email dataframe, I tokenized and removed stop words in order to understand the frequency distribution of words found in phishing emails. I also constructed a word cloud to better understand common themes.

I then tokenized combined dataframe and removed stopwords. I then performed TF-IDF vectorization, using a pipeline that contained the vectorization and TF-IDF steps in combination with the classifier of choice. My baselines consisted of Random Forest Classification and the SEAHound method. The SEAHound method's results are publicly available. However, for the Random Forest baseline, I performed 5-Fold Cross Validation on the data and accounted for average accuracy, precision, and recall.

This same process was repeated using a Multinomial Naive Bayes classifier and a Support Vector Machine Classifier, and with the Netcraft link analysis step. The classifier of choice, based on performance, was the Support Vector Machine classifier with Stochastic Gradient Descent learning.

5 Evaluation and Results

My evaluation metrics include precision, recall, and accuracy. High precision is important because this tells us the percentage of the phishing emails filtered out that are actually phishing. Precision is crucial to maintain so important, non-phishing emails don't get filtered out. Recall is arguably the most important metric because limiting phishing attempts from reaching employee inboxes is the number priority. 100% recall means that zero phishing emails were able to properly reach their target. Accuracy is an added metric which is important in evaluating the success of a classifier. Using 5-Fold cross validation on each classifier (excluding SEAHound), including iterations with the Netcraft step and without, I was able to extract average precision, recall, and accuracy for each system.

[2] https://github.com/tianruip1994/Semantic_Analysis

[3] https://www.netcraft.com

Method	Precision	Recall	Accuracy
Random Forest	**100%**	84%	92%
SEAHound	95%	91%	N/A
MNB	**100%**	94%	97%
N+SVM+SGD	**100%**	**100%**	**100%**

As depicted by the table, my method of Netcraft + Support Vector Machine + Stochastic Gradient Descent outperforms other methods in terms of precision, recall and accuracy. This is clearly the preferred system for phishing email classification.

6 Discussion

My system is overwhelmingly successful in classifying phishing emails. It significantly outperforms baselines, highlighting the success of the system. Not only is it relatively high performing; however, it is almost a perfect classification system on an absolute scale, which leads me to attribute this to the diversity in data sources. In the future, I would love to test the system on emails from the same source. Additionally, while the 2 datasets I concatenated are very similar in their format, there are some differences which may need some fine tuning. If the model learns to spot the differences in datasets based on format rather than the actual text, this may lead to a false positive in my study.

The random forest baseline performed better than expected, resulting in a very high precision. However, its recall is substantially underperforming when compared to the other methods. I had expected the Multinomial Naive Bayes method to outperform all of the others, which is why I was pleasantly surprised when the Support Vector Machine surpassed its performance in nearly every metric.

7 Conclusion

This phishing email filtering system of link verification in combination with TF-IDF vectorization and Support Vector Machine classification is the best performing in regards to this dataset. While I'm confident I did not overfit the data, I cannot make the claim that this is the best phishing detection system system due to the lack of labeled data from a singular source.

However, I can conclude that this system is a very efficient means of sorting phishing emails. This system by itself can serve as an effective phishing monitoring tool, of which its methods can be adopted for companies and email service providers alike [4].

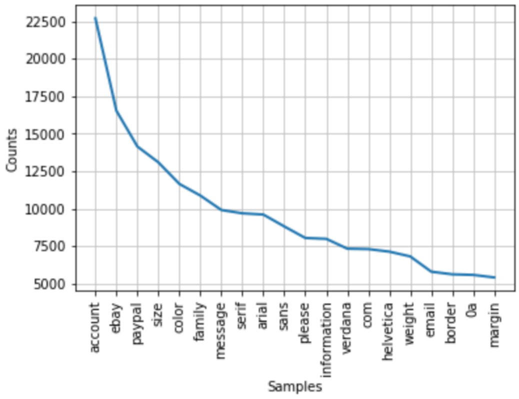


Figure 3: Phishing Email Frequency Distribution

8 Next Steps

The most important next step is to demonstrate this system's effectiveness on another data source. The non-phishing and phishing datasets come from completely different email sources. There are a multitidue of problems with this, which include the types of emails being exchanged as well as the format. I believe an appropriate solution is to use a singular email dataset for both the phishing and non-phishing emails. For this, crowdsourcing the labeling will be necessary, and I recommend hiring work from Amazon Mechanical Turk for the labeling portion.

Additionally, in exploring the frequency distribution of tokenized phishing emails, common words were "account, ebay, paypal", as well as unexpected words like "color" and "font", which highlights improvements that ought to be made in scraping HTML text data from the mbox email files.

It is unknown whether this system will be as effective on other email datasets. Which is, again, why I believe it is important to find a new labeled email dataset from a singular source. My hypothesis is that my system won't perform as well on an absolute performance standpoint, but it will still outperform the other systems when tested on a new dataset.

[4] https://github.com/shaeferd/Phishing-Detection

References

2018. Global spam volume as percentage of total email traffic from January 2014 to December 2018.

Clarence Chio and David Freeman. 2018. *Machine Learning and Security*, volume 1. O'Reilly, Sebastopol, CA.

Lorrie Cranor, Serge Egelman, Jason Hong, and Yue Zhang. 2006. Phinding Phish: An Evaluation of Anti-Phishing Toolbars. *CyLab Carnegie Mellon University* .

Rohit Giyanani and Rohit Giyanani. 2014. Spam Detection using Natural Language Processing. *IOSR Journal of Computer Engineering* .

Bryan Klimt and Yiming Yang. ???? The Enron Corpus: A New Dataset for Email Classification Research. *Language Technologies Institute Carnegie Mellon University* .

Tianrui Peng, Ian G. Harris, and Yuki Sawa. 2018. Detecting Phishing Attacks Using Natural Language Processing and Machine Learning. *2018 12th IEEE International Conference on Semantic Computing* .