Evaluating a Semisupervised Approach to Phishing URL Identification in a Realistic Scenario

Binod Gyawali, Thamar Solorio, Manuel Montes-y-Gómez, Bradley Wardman, Gary Warner
Department of Computer and Information Sciences
University of Alabama at Birmingham
1300 University Blvd.
Birmingham, Alabama, USA
{bgyawali,solorio, mmontesg, bwardman, gar}@cis.uab.edu

ABSTRACT

Phishing sites have become a common approach to steal sensitive information, such as usernames, passwords and credit card details of the internet users. We propose a semisupervised machine learning approach to detect phishing URLs from a set of phishing and spam URLs. Spam emails are the source of these URLs. In reality, the number of phishing URLs received through these spam emails is fewer compared to other URLs. Our study is targeted to detect phishing URLs in a realistic scenario of a highly imbalanced data set containing phishing and spam URLs with 1:654 ratio. To train a learning algorithm labeled URLs are needed, where manual labeling is a common approach. Given that it is not feasible to manually label all the URLs from large data sets, we propose reducing manual intervention by labeling only 10% of the URLs manually and using a semisupervised learning algorithm. We compare the proposed approach with a supervised learning approach. Evaluation results show that our proposal is competitive if it is applied in combination with appropriate feature selection and undersampling techniques.

Keywords

Phishing URLs Identification, Semisupervised Learning, Imbalanced Data

1. INTRODUCTION

With the increasing use of the Internet for online transactions, criminals have become very active in stealing sensitive information using phishing websites. Phishing websites cheat users by making them feel that they are in a secure website by exploiting the use of URLs and design of web pages. In most cases, users receive emails with links to webpages where the true URL may not be seen until the link is clicked. Users are unable to judge by looking at these links if they are redirected to a safe website or not.

Global Phishing Survey[5] shows that there were at least 67,677 phishing attacks worldwide in the second half of 2010 and shows the number of attacks is increasing. Phishing Trend Report[4] shows that in the second quarter of 2010, the most targeted sectors are payment services with 38%, financial services with 33% and classifieds having 6.6% of total phishing attacks. It also shows that the United States still remains the top country to host phishing sites. Thus, efficiently identifying phishing URLs has become a great necessity and challenge in the present context.

The basic approach for phishing URL detection is the blacklisting approach. Blacklisting works on the basis of a pre-compiled database of URLs those at some point were found to be phishing sites. The database may not be updated with new phishing URLs and thus may become obsolete very quickly. In order to alleviate this problem, several machine learning techniques are applied to detect the phishing URLs on the fly. These automatic techniques deal with phishing URL detection as a classification problem. Most of the previous phishing URL identification experiments apply a supervised learning approach and deal with a small and balanced data set[9][13][14]. When a supervised machine learning technique is used for the classification, it requires large amounts of training data that is manually labeled. When the data set is very large, it is not feasible to manually label the URLs, as the cost of labeling is very high. Our study is focused in reducing the cost of training a supervised algorithm by relying on fewer manually labeled data. In order to reduce the manually labeled data, we propose to apply a semisupervised approach and train the learning algorithm by a collection of manually labeled and pseudo labeled data. We show that this approach of using only 10% of manually labeled data is able to detect phishing URLs comparative to a fully supervised approach.

When any classification algorithm is applied to an imbalanced data set, the classifier tends to be biased towards the majority class[7][10][12][19]. The problem of classification becomes worse when the minority instances are very few in numbers. In a realistic scenario, the number of phishing URLs is never equivalent to the spam URLs. Rather, the number of phishing URLs is fewer compared to spam URLs. Our work is directed towards detecting phishing URLs from a realistic occurrence of a highly imbalanced data set.

Thus, our overall approach is focused on two different studies: i) reduce the manually tagged data and ii) over-

CEAS 2011 September 1-2, 2011, Perth, Western Australia, Australia.

come the problem of an imbalanced data set. Our proposed approach is able to identify phishing URLs with 7.5% Cumulative Error Rate. Our study is different from previous work in many aspects including applying a semisupervised approach, using only lexical features generated from URLs, a realistic distribution of spam and phish data with an average ratio of 1:654 and a diverse data set containing a very broad feed of phishing URLs targeting 392 different brands.

The remainder of this paper is organized as follows. Section 2 discusses previous related work. Section 3 gives the overview of our proposed methodology and the feature selection method to train the model. Section 4 covers the data set details. Section 5 describes the data selection approach and performance evaluation methods. Section 6 covers the results and its evaluation. Finally, Section 7 covers discusses our findings and final remarks.

2. RELATED WORK

The characteristics of phishing URLs are different from the legitimate URLs. McGrath et al.[15] study by comparing the phishing URLs from PhishTank and MarkMonitor with the regular URLs from DMOZ and show that the phishing URLs are shorter in length than the regular URLs. They show that 50-75% of phishing URLs contain the name of the brand that they target either in top level domain or in the path of URLs. The domains of phishing URLs also tend to use fewer vowels and contain fewer unique characters.

The basic technique of phishing URL detection is through the use of antiphishing toolbars in the browsers. These come as an extension of web browsers and warn users when they are visiting a suspected phishing site. Nevertheless, these do not offer a high level of protection, and are not always up to date. Wu et al.[18] present a study on five different toolbars, SpoofStick, Netcraft Toolbar, Trustbar, eBay's Account Guard, and SpoofGuard, and show that security toolbars are ineffective in preventing phishing attacks since the users fail to pay attention on warning given by the toolbars.

Based on the types of features used for the classification of URLs, there are mainly three different approaches using machine learning. They are: using content based features, lexical features, and host based features. Some of the studies also use a combined approach of using lexical and host based features for phishing detection.Wenyin et al.[17] present the content based approach of phishing webpage detection. This approach first decomposes web pages into salient block regions and evaluates visual similarities between benign webpages and phishing webpages. The similarity is calculated in three parts, viz. block level similarity, layout similarity, and overall style similarity. Though this approach works well, detecting webpages using visual similarity may not be too adequate for a real time classification and it may be harmful to download the phishing webpage. This approach is more useful for a true webpage owner who wants to detect phishing URLs targeted to his webpage.

Work of Ma et al.[14] demonstrates the detection of malicious websites using both lexical and host based features of URLs in a balanced set. They use a data set of 20,000 URLs per day having an average benign to malicious URLs ratio of 2:1 and applies various online algorithms such as Perceptron, Logistic Regression with Stochastic Gradient Descent, Passive Aggressive and Confidence Weighted algorithms to compare with a batch processing algorithm (Support Vector Machine), and demonstrates that online learning algorithms work better than batch learning for detecting malicious websites. Another similar work is Blum et al.[9] that focuses on classification of URLs based on lexical features using online learning.

Le et al.[13] show that lexical features of URLs are sufficient for the classification of URLs. They use five different combinations of data sets collected over a short duration of time, the largest data set having 6083 phishing and 8155 non phishing URLs. They compare several learning algorithms and propose to use an Adaptive Regularization of Weights (AROW) method that is able to overcome noisy training data. Similarly, Nimeh et al.[6] use a data set of 2889 phishing and legitimate emails with 59.5% legitimate emails to compare machine learning techniques in predicting phishing emails. Among several machine learning methods, they propose that, the learning methods are dependent on the problems and must apply measures that best fit the problems.

To the best of our knowledge, there has been no study on detecting phishing URLs that deals with a skewed distribution of instances that is expected in realistic conditions. Most of the above mentioned approaches deal with a small and balanced data set. Though online learning approach is more dynamic than batch learning and is faster in execution, both will suffer with imbalanced data sets. Moreover, these approaches require all the training data to be prelabeled. When the data set is large, the cost of labeling all the instances in it is very high. Thus, the above mentioned approaches become infeasible in a large and imbalanced data set.

3. PROPOSED METHODOLOGY

Similar to previous works, we deal with phishing URL detection as a classification problem. We apply a semisupervised approach of learning to train the classifier to detect the phishing URLs. We conduct training and testing on a daily basis. For each day of testing, data from previous days is used to train the classifier.

Section	Example
URL	http://update.paypal.com.3dx0v1k47fiu95-dsn7m37s23a52c25g26m3dx2sv5u1x1.3dx0-v1k47f1pm4vf7u95de2n7m3k471p52m2iu5t-3xc3a0a3k7f1pm4vf7u34de.autoextraparts-.com/cgl_bin/webscr.html?cmd=5885d80-a13c0db1f22d2300ef60a67593b79a4d0374-7447e6b625328d36121a1e7043d426372b26-4a16877a137a6684ae7043d426372b264a16-877a137a6684a
Scheme	http
Authority	update.paypal.com.3dx0v1k47fiu95dsn7m3-7s23a52c25g26m3dx2sv5u1x1.3dx0v1k47f1-pm4vf7u95de2n7m3k471p52m2iu5t3xc3a0-a3dk7f1pm4vf7u34de.autoextraparts.com
Path	cgl_bin/webscr.html
Query	cmd=5885d80a13c0db1f22d2300ef60a6759-3b79a4d03747447e6b625328d36121a1e704-3d426372b264a16877a137a6684ae7043d42-6372b264a16877a137a6684a

Table 1: Parts of a URL

Since the number of negative instances is very large com-

General Features (17)	Domain Features (11)	Path Features (10)	Query Features (9)
Length of URL	Top Level Domain	All Path	Query0
URL Dot Count	Domain1	Last Path	Query1
URL Token Count	Domain2	Path1	Query2
Domain Length	Domain3	Path2	Query3
Domain Dot Count	Domain4	Path3	Query4
Domain Hyphen Count	Remainant Domain	Path4	Query1.Query0
Domain Token Count	Domain1.Domain0	Path1.Path0	Query2.Query1
Path Length	Domain2.Domain1	Path2.Path1	Query3.Query2
Path Token Count	Domain3.Domain2	Path3.Path2	Query4.Query3
File Name	Domain4.Domain3	Path4.Path3	
File Extension	All Domain		
Query Length			
Query Tokens Count			
Is IP Address			
Valid Domain Tokens Count			
Port			
Protocol			

Table 2: Features of URL

pared to the number of positive instances, we undersample the training data to make the number of negative instances comparable to the number of positive instances. From the set of positive and undersampled negative instances, we select relevant features from all the URLs to create feature vectors and use them to train the classifier. We also apply the feature reduction techniques to improve the performance that are discussed in section 3.2.

We use Support Vector Machine Light (SVM Light) as the classifier[11] in our task.

The sections below describe the format of the URLs, their parts, and the way we represent them.

3.1 Parts of a URL

According to RFC1738 standard[3], a URL can be broken down into the following parts.

<scheme>://<authority:port>/<path>?<query>

We see that the scheme is separated by ":" from the rest of the URL. After scheme, URLs contain authority, which is the domain of the URL and is separated from scheme by "//". Similarly, authority and path are separated by "/" and path and query are separated by "?".

Tokens within authority are separated by ".". Path tokens are separated by "/". Query tokens are separated by either ";" or "&". Table 1 shows the division of a URL into its various parts.

3.2 Feature Selection

In our project, 47 different feature types are created from the tokens of different parts of a URL, and the features are represented by a Bag of Words representation. A URL is divided into four parts: scheme, authority, path and query. Features are extracted from each of these parts. According to the information content of the feature and the parts from where they are generated, the features are divided into four groups.

- **General Features:** There are seventeen general features that are related to the structure of URLs including the length of URL, number of tokens, whether it contains an Ip-address or not, protocol used etc. We also use a feature to know the number of valid dictionary words contained in a domain. This feature is named as Valid Domain Token count. These features are listed in Table 2 under column 1.

- **Domain Features:** There are 11 features that are generated from the domain of a URL. Domain of a URL is split into parts by "/", "?", "=", ",", "-", "_", "&" and "." characters and various combination of these parts are used to generate domain features. The list of features are shown in the second column of Table 2.

- **Path Features:** This group of features includes 10 features related to the unigram and bigram of path. Path of a URL is split into parts by "/" and are used to generate path features of URL. Details of path features are shown in the third column of Table 2.

- **Query Features:** These features are related to the query of a URL. A query is split into parts by ";" or "&" and unigrams and bigrams of these tokens are used to generate query features. There are 9 features generated from queries. These features are shown in the fourth column of Table 2.

Binary weight, i.e. 0 or 1, is assigned to each feature. To reduce the data sparseness, we removed the features that are present only once in the whole data set.

4. DATA SET

The data set used in this research was collected from November 1^{st} 2010 to January 31^{st} 2011. There are two categories of URLs within the data set: phishing and spam. The set of 65,855 phishing URLs were obtained from a trusted research partner. These URLs were processed by the anti-phishing company's system and labeled as phish. Many of these URLs were manually reviewed to ensure the accuracy of the labels and the number of false positives in this data is less than 1%. The set of 43,086,508 spam URLs were extracted from the same date range of spammed email messages in the UAB Spam Data Mine. M86 Security Labs[1] claims that less than 0.01% of spammed URLs are phish. MessageLabs[2] gives a slightly higher number, closer to

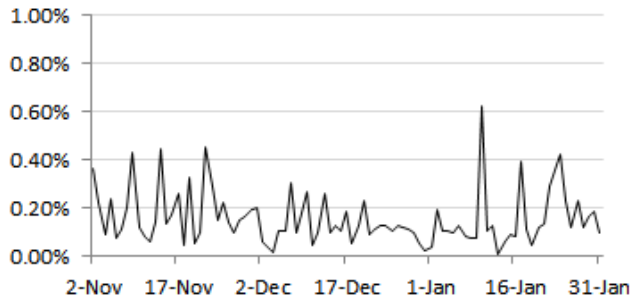

Figure 1: Percentage of Phish to Spam ratio

0.25%. Therefore by taking into account the number of removed URLs from the spam set, there should be a low number phishing URLs remaining in the spam URL set, and thus, we assume this has a small effect on the false positive rate. The ratio of phish to spam URLs is shown in Figure 1.

Our data set consists of phishing URLs that are targeted for 392 different brands. Of the 392 different target brands, there are 131 target brands having one phishing URL for each. Similarly, the next 42 brands have only 2 phishing URLs targeted to them. We consider this phish set a very diverse set and argue that this is much more diverse to phish used in previous works[1]. The most targeted brands in our data set are shown in Table 3.

Target Brands	Phishing URLs
Paypal	20409
Chase Bank	5003
Wells Fargo	4912
HSBC	3907
Bank of America	3432

Table 3: Most targeted brands by phishers

5. EXPERIMENTAL SETUP

In this section, we discuss our data selection approach to train the classifier and the performance evaluation methods.

5.1 Data Selection and Undersampling

Our data set is highly imbalanced and is dominated by the non-phish class, which is a great challenge in automatic data classification. A number of approaches have been used to address the imbalanced data problem. The commonly used techniques are: undersampling majority instances or oversampling minority instances to balance the training set. There are also approaches like feature selection in which the features with high information and most indicative of membership are selected and used for the classification[19]. In a feature selection approach, the goal is to find the value of each feature, selecting only those features that contain high association with the class. But, when the number of features is very large, finding information of features becomes an expensive overhead and requires high processing time.

Chawla et al.[10] propose the Synthetic Minority Oversampling Technique (SMOTE) algorithm to oversample the minority class. This technique creates new minority class instances that lie between an instance and its neighbors using k nearest neighbors of existing instances and their features. Akbani et al.[7] use SVMs for the classification and apply SMOTE with Different Costs (SDC) approach, which is the combination of SMOTE and using different error costs for positive and negative instances. One of the disadvantages of these approaches is that it adds more instances in the existing data set and increases the data set which indeed increases the processing time. Therefore, when the number of instances is large, oversampling is not a good approach. Similarly, Kubat et al.[12] use one sided selection (undersampling) approach where all the minority class instances and a subset of instances from the majority class are selected to train the classifier.

Because of the very large number of instances in our data set, our study uses an undersampling approach as in[12], which reduces the number of negative instances and make them equivalent to positive instances to train the classifier.

In our experiments, we use 15 days of data prior to the test day to train the classifier. This is chosen since it gave us the best results in our previous experiments.

5.1.1 *Supervised Learning*

Training in supervised learning is done by using only the manually labeled URLs. To train the classifier in this approach, we apply undersampling to select the spam URLs and upsampling of the phishing URLs. We select phishing URLs from the last 15 days before the test day and spam URLs from only one day before the test day. Spam URLs are selected by random undersampling such that number of spam URLs is twice the total number of distinct phishing URLs of 15 days. Figure 2 shows how we select the training data on 25^{th} day of the experiment.

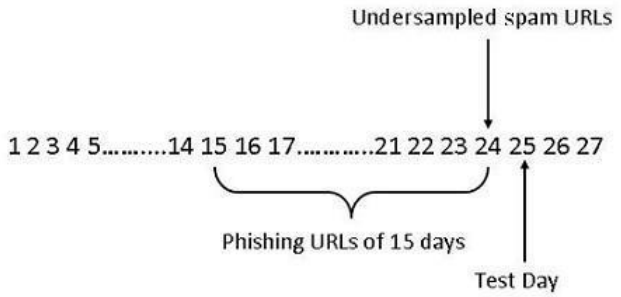

Figure 2: Training data selection in supervised learning

All the phishing URLs are upsampled by using each of the phishing URLs twice to create the feature vector. In feature selection, we remove those features that are present only once to reduce the data sparseness. Thus, upsampling of phishing URLs is done to preserve all the features of the phishing URLs.

5.1.2 *Semisupervised Learning*

In semisupervised learning, manually labeled as well as unlabeled data are used to train the classifier. A classifier is first trained with labelled data and tested with a subset of unlabeled data. The label predicted by the classifier on the previously unlabeled data is now used in combination with

[1]Work by Blum et al.[9] states that their phishing URL set is targetted to 177 different brands

the manually labeled data to train a new classifier. The data labeled by the classifier and used for training are referred as pseudo labeled data.

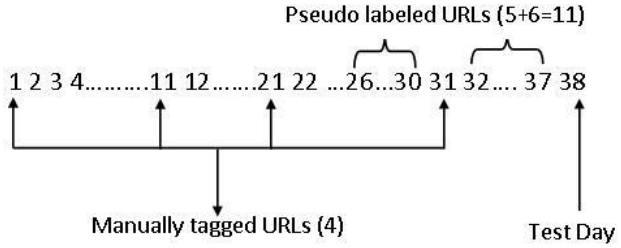

Figure 3: Training data selection in semisupervised learning

In this approach, we use these manually labeled as well as pseudo labeled data to train the classifier. Moreover, we apply only undersampling of data in this approach. In this approach, we have labeled URLs in each 10 days interval. To test URLs on any day, we select the latest maximum 7 days labeled URLs and the latest minimum 8 days of pseudo labeled URLs, apply undersampling in each days' data and use this set in training the classifier. Thus, the total data used for training will always be of 15 days. But, the proportion of manually labeled and pseudo labeled data may be different depending upon the day when the testing is done. As in Figure 3, to test URLs on 38^{th} day of experiment, there are only 4 days of manually labeled URLs and the remaining 11 days of pseudo labeled URLs. But, to test data on the 87^{th} day, we have the latest 7 days of manually labeled URLs and the latest 8 days of pseudo labeled URLs.

For manually labeled data of each day, undersampling of spam URLs is done at random such that the spam and phishing URLs are equal. For pseudo labeled data of each day, we first order the positive and negative classified instances by their confidence value and select the highest confident data such that the number of positive and negative instances each are at most 1200.

5.2 Performance Evaluation Metrics

The commonly used performance measures in text classification are Precision, Recall and F1 measures. These measures work well in a balanced data set but are not a good way to measure performance of imbalanced data classification[12]. In an imbalanced data set, a classifier always tends to be biased towards majority class. These performance measures give equal importance to the misclassification made on positive and negative instances. In a highly imbalanced data set, it is better to give different weights to the misclassification of positive and negative instances[8]. Thus, we use Balanced Success Rate (BSR) to measure the performance of our approach. Balanced Success Rate is the arithmetic mean of specificity and sensitivity as given in Equation 1.

$$BSR = \frac{Speicificity + Sensitivity}{2} \quad (1)$$

Specificity and sensitivity are the statistical measures where sensitivity measures the proportion of positive instances that are identified correctly, while specificity measures the proportion of negative instances that are identified correctly, out of the total numbers of instances in the test set.

$$Sensitivity = \frac{True\,Positive}{True\,Positive + False\,Negative} \quad (2)$$

$$Specificity = \frac{True\,Negative}{False\,Positive + True\,Negative} \quad (3)$$

We also use Cumulative Error Rate (CER) to measure the percentage of phishing URLs that are misclassified. Cumulative Error Rate is the ratio of cumulative sum of the number of false negatives to the cumulative sum of the number false negatives and true positives. Cumulative Error Rate on the n^{th} day is calculated as shown in equation 4 .

$$CER = \frac{\sum_{i=1}^{n} False\,Negative_i}{\sum_{i=1}^{n}(False\,Negative_i + True\,Positive_i)} \quad (4)$$

6. EXPERIMENTAL RESULTS

The Cumulative Error Rate(CER) and Balanced Success Rate(BSR) of the experiment done for three months of data are shown in Figures 4 and 5 respectively. From these figures, we see that there is a very small difference in Cumulative Error Rate and Balanced Success Rate of supervised and semisupervised learning.

In Figure 4, we see the cumulative error rate of the two methods are almost the same towards the end of the experiment. But, in the beginning of the experiment, semisupervised learning has a greater error rate. This is because in the initial days, the number of labeled data is smaller and the classifier could not generalize well. But towards the end, the labeled data used for training goes on increasing, and this helps improving the classification accuracy. Thus, towards the end of experiment, we see the performance of these two methods are almost equivalent. This shows that, instead of applying a fully supervised method, application of the semisupervised learning approach helps the classifier reach performance almost equivalent to the supervised approach and greatly reduces the manually labeled data.

In Figures 4 and 5, we see a sudden change on 42^{nd} (13^{th} December) and 77^{th} (17^{th} January) day of the experiment. This change is not seen in the semisupervised approach on the 42^{nd} day as the classifier was not well trained until this day. This sudden rise in CER and fall in BSR is seen due to a high false negative rate. Further investigation found that the technique is unable to detect a couple of large sets of URLs that contain different domains but all have the exact same path. These URL sets, hereafter referred to as "tilde phish" are the results of web servers configured to allow a file path to be shown on any of the virtual domains hosted on that server. On a web server hosting both "domainxyz.com" and "domainabc.com", a phish may be placed at the server path /∼user/myphish/. Every domain on that server would now be capable of observing this phish, making "domainxyz.com/∼user/myphish/" and "domainabc.com/∼user/myphish/" appear as valid phishing sites. These additional virtual server phishing URLs are not being observed in the wild, but are being reported through at least one phishing feed provider, which is propagating through many users of that feed. In servers with large numbers of virtual domains, there may be more than 1,000 URLs all sharing the same URL path, which can cause an artificially inflated value to be assigned to those paths.

Historically another example of "over-counting" exists in previous data sets containing "Rock Phish" URLs. The crim-

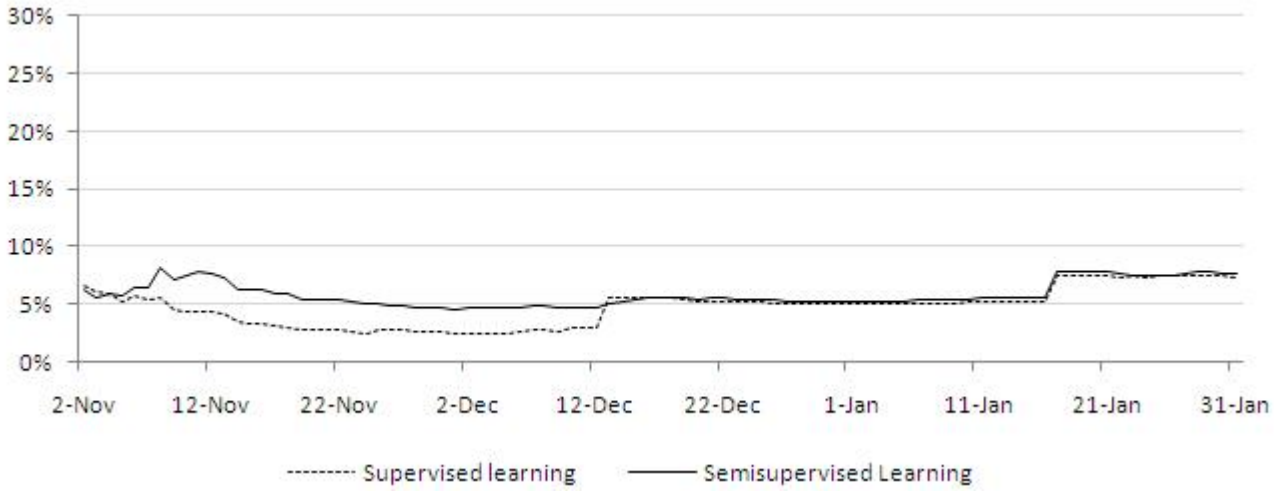

Figure 4: Cumulative error rate in supervised and semisupervised learning

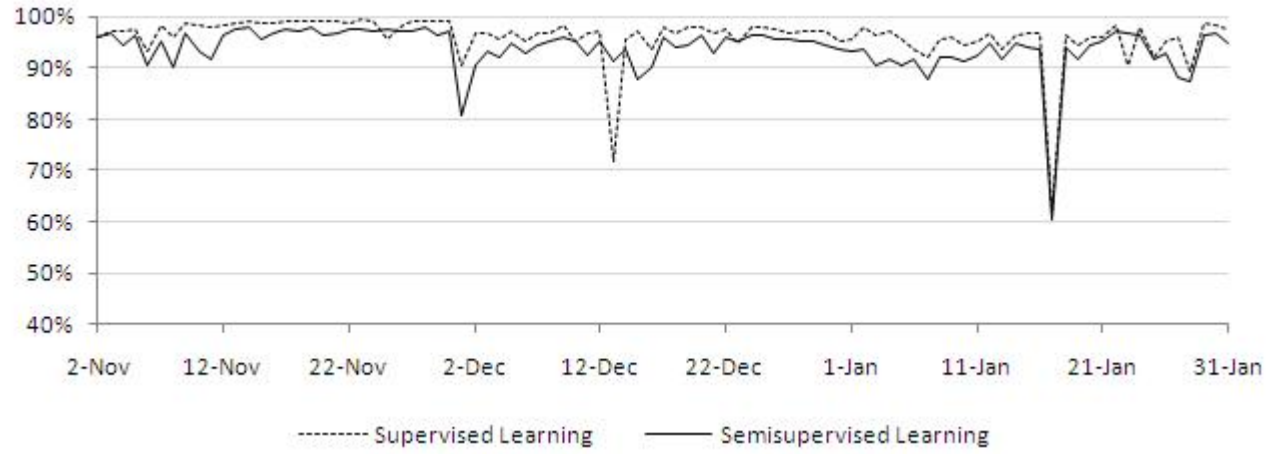

Figure 5: Balanced success rate in supervised and semisupervised learning

inal group known as Rock Phish throughout the industry created many randomly generated machine names on the same primary domain that could all point to one phishing website and were at one point in time said to have been behind around 50% of phishing websites[16]. Because of the randomly created URLs, a single domain could be seen to have tens of thousands, or even hundreds of thousands of URLs, all of which resolved to phishing sites. Again, machine learning techniques would naturally assign an overly strong weight to features found in these domain names, and this trend is noted in several previous research papers. To provide an accurate view of the true risk, another experiment is run using a subset of phishing URLs, 37,071, not including tilde phish to show how these types of URLs may or may not have an effect on the error rate[2] applying semisupervised learning using pseudo labeled data.

This experiment is run using the semisupervised approach of learning in the original dataset and the dataset removing the tilde URLs. The Cumulative Error Rate and Balanced Success Rate for the experiment are in Figures 6 and 7 respectively. Figure 6 shows that, the cumulative error rate reduces greatly when we remove tilde URLs and there is no sudden rise in the cumulative error rate on 42^{nd} and 77^{th} day of the experiment. Thus, we can say that, the abrupt appearance of tilde phish on any day makes it difficult to identify the phish and results to rise in the detection error rate. Similarly, in Figure 7, we see that there is no sudden decrease in BSR on the 42^{nd} and 77^{th} day.

The smaller number of URLs available for training in tilde removed data set causes the rise in false positives. Thus the Balanced Success Rate of tilde removed data set is fewer than in the original data set. The BSR curve in Figure 7 shows there is a decrease in BSR on the 1^{st} of December and 1^{st} of January but there is no abrupt change in cumulative error rate. This drop is seen in both the original and tilde removed data set though the effect is less on 1^{st} of January in the original data set. This decrease in BSR on these days is due to the increase on the false positive rate. We obtained our spam data set in three different bundles of three months.

[2]Previous researchers have noted the presence of such URLs, but have not shared the results over their techniques if the data is removed.

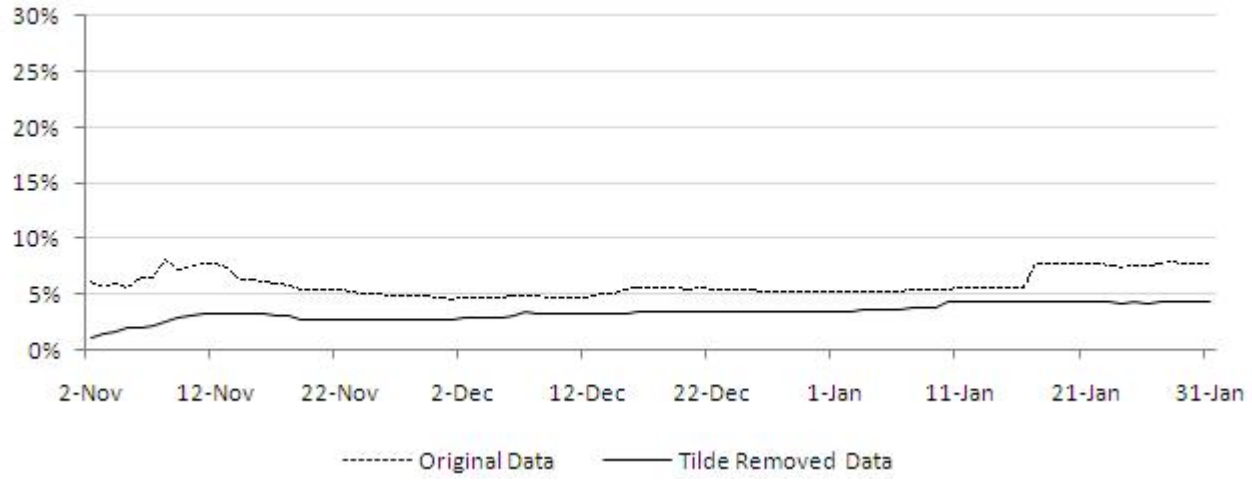

Figure 6: Cumulative error rate in original and tilde removed data using semisupervised learning

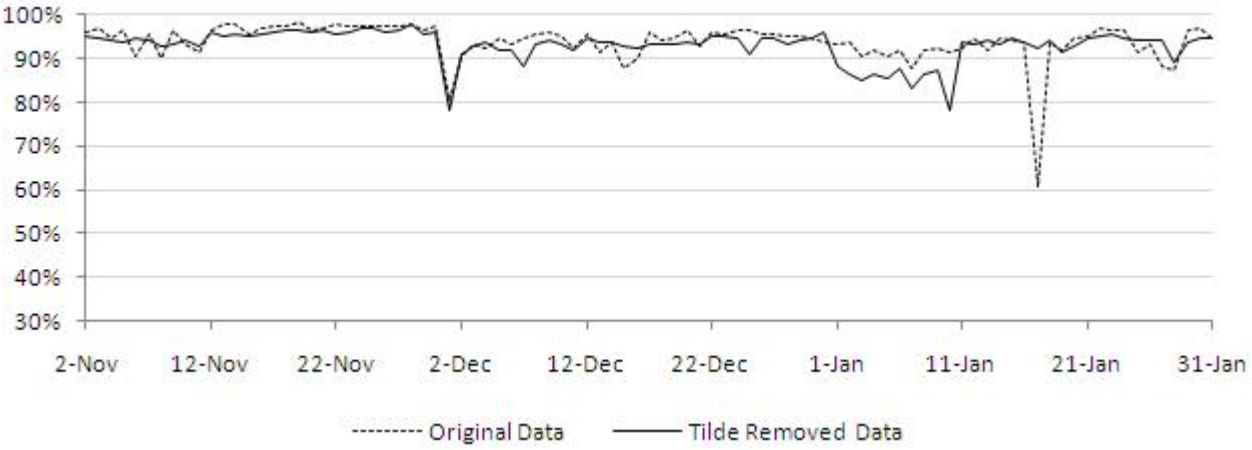

Figure 7: Balanced success rate in original and tilde removed data using semisupervised learning

So, there may be a sudden change in the type of spam URLs that we received. Due to the abrupt change in the spam URLs in different months, the classifier trained by using previous months' data is unable to detect the spam URLs in the start of new months. After the 1^{st} of December, the BSR curve tends to rise which shows that when the classifier is trained with the data of 1^{st} of December, it is able to detect spam URLs. But, the BSR curve is still unable to rise after the 1^{st} of January until 11^{th} of January. This is due to the use of pseudo labeled URLs for training. On the 11^{th} of January, the classifier is trained by baseline URLs of the 10^{th} of January, so it is able to detect spam URLs more properly. Moreover, the number of phishing URLs after removing tilde URLs is reduced to almost 50% of the original ones. So, this reduces the number of URLs used for training and hence, the classifier could not be well trained due to very few training data. Thus this is seen more in tilde removed data set than in the original data set.

Going through the results of these experiments, we discovered the following kinds of URLs that are misclassified:

1. Some very short URLs like "http://epoqdpu.tk/" are misclassified because of the small number of features present in it. There are some URLs that have Ip-address present in them. These types of URLs are also misclassified.

2. Some of the very long spam URLs having a greater number of features are misclassified. The reason behind the misclassification might be due to random undersampling which might have excluded URLs with very high information from the training. So, these types of URLs are misclassified.

3. There are shortened URLs such as "http://x.co/L4Qn" present in the data set which are also not classified correctly. These shortened URLs do not have any information in them. Similarly, there are many spam URLs that are invalid. These invalid URLs when selected for training might have affected the classifier and reduced the performance of classifier.

7. CONCLUSION

This paper presents a study on automatic phishing URL identification under realistic conditions, where the data is highly imbalanced and the phish data set is very diverse. Previous work have presented results with artificially balanced sets and proposed to use learning algorithms that require very large amounts of labeled data, which is highly expensive. Our study attempts to answer the following two questions:

How much we can reduce manually tagged data: We show a comparative analysis of a supervised learning using all the manually tagged data and a semisupervised learning using only the 10% manually tagged data to identify the phishing URLs. The experiment shows that the semisupervised approach is able to identify phishing URLs comparative to using the supervised approach with a 7.5% Cumulative Error Rate.

How we can overcome the problem of imbalanced data: We use a highly imbalanced dataset having a realistic distribution of phishing and spam URLs with ratio 1:654 and a very diverse phishing data set targeted to 392 different brands. Applying an undersampling technique and an appropriate feature selection and data selection method is able to alleviate the problem.

Our approach is acceptable because of its good accuracy in detecting phishing in an imbalanced data set and is able to reduce manually tagged URLs up to 10% of the total URLs.

As future work, we could to evaluate other techniques for undersampling. One of them may be applying clustering approaches to group similar URLs and select URLs from all the clusters. This helps us to include diversity of URLs and their features in training and may help in improving the performance. Future work could also involve using feature selection strategies to identify the most important features of URLs and train the model. This helps us to reduce the feature set and helps to improve the performance and efficiency.

8. REFERENCES

[1] http://www.m86security.com.

[2] http://www.messagelabs.com.

[3] Rfc1738- internet engineering task force.

[4] Phishing activity trends report second quarter 2010. *Anti-Phishing Working Group (APWG)*, 2010.

[5] G. Aaron and R. Rasmussen. Global phishing survey: Trends and domain name use in 2h2010. *Anit-Phishing Working Group (APWG)*, April 2011.

[6] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair. A comparison of machine learning techniques for phishing detection. In *Proceedings of Anti-Phishing Working Group's eCrime Researchers Summit*, pages 60–69, Pittsburgh, PA, USA, 2007.

[7] R. Akbani, S. Kwek, and N. Japkowicz. Applying support vector machines to imbalanced datasets. In *Proceedings of the 15th European Conference on Machine Learning (ECML)*, pages 39–50, Pisa, Italy, 2004.

[8] A. Ben-Hur and J. Weston. *A User's Guide to Support Vector Machines*. Biological Data Mining. Oliviero Carugo and Frank Eisenhaber (eds.) Springer Protocols, 2009.

[9] A. Blum, B. Wardman, T. Solorio, and G. Warner. Lexical feature based phishing url detection using online learning. In *Proceedings of the 3rd ACM workshop on Artificial Intelligence and Security*, New York, USA, 2010.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, November 2002.

[11] T. Joachims. *Making large-Scale SVM Learning Practical*. Advances in Kernel Methods - Support Vector Learning , B. Scholkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

[12] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997)*, pages 179–186, Nashville, Tennessee, USA, July 1997.

[13] A. Le, A. Markopoulou, and M. Faloutsos. Phishdef: Url names say it all. In *Proceedings of the 30th IEEE INFOCOM*, Sanghai, China, April 10-15 2011.

[14] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Identifying suspicious urls: An application of large-scale online learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 681–688, Montréal, Québec, Canada, June 2009.

[15] D. K. McGrath and M. Gupta. Behind phishing: An examination of phisher modi operandi. In *First USENIX Workshop on Large-Scale and Emergent Threats*, USA, 2008.

[16] R. McMillan. "rock phish" blamed for surge in phishing. *InfoWorld*, December 12th 2006.

[17] L. Wenyin, G. Huang, L. Xiaoyue, X. Deng, and Z. Min. Phishing webpage detection. In *Proceedings of 8th International Conference on Document Analysis and Recognition (ICDAR'05)*, pages 560–564, Seoul Korea, 2005.

[18] M. Wu, R. C. Miller, and S. L. Garfinkel. Do security toolbars actually prevent phishing attacks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 560–564, Montréal, Québec, Canada, April 22-27 2006.

[19] Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, 6, June 2004.