Journal of Theoretical and Applied Information Technology
30th June 2016. Vol.88. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

LEXICAL BASED METHOD FOR PHISHING URLS DETECTION

[1]AMMAR YAHYA DAEEF, [2]R. BADLISHAH AHMAD, [3]YASMIN YACOB

[1,2,3]School of Computer and Communication Engineering, Universiti Malaysia Perlis (UniMAP), Perlis, Malaysia
[1]Middle Technical University, Baghdad, Iraq
E-mail: [1]ammaryahyadaeef@gmail.com, [2] badli@unimap.edu.my, [3]yasmin.yacob@unimap.edu.my,

ABSTRACT

Phishing is a social engineering attack that exploits user's ignorance during system processing has an impact on commercial and banking sectors. Numerous techniques are developed in the last years to detect phishing attacks such as authentication, security toolbars, blacklists, phishing emails, phishing websites, and URL analysis. Regrettably, nowadays detection system implemented for specific attack vectors such as email which make developing wide scope detection is much needed. Previous studies show that analysis of URLs proved to be a good option to detect malicious activities where this method mostly based on features of lexical, host information, and other complex method which requires a long processing time. In this paper, we present phishing detection system using features extracted from URLs lexical only to meet two important goals which are wide scope of protection and applicability in a real-time system. The system provides accuracy of 94% and can classify single URL in average time of 0.12 second.

Keywords: *Phishing, Classifiers,* Machine *Learning, Lexical Features.*

1. INTRODUCTION

Phishing is a cutting edge threat that has an impact on commercial and banking sectors by means of the Internet which delivers huge misfortunes at the level of clients and organizations [1]. Phishing can be characterized generally as Internet fraud which is done through the utilization of social engineering with the end goal to acquire vital and touchy data of users such as passwords or a credit card for illicit purposes by utilizing several vectors of attacks, for example, the websites, advertising on websites, emails or even through phone calls [2]. In spite of the broad field of phishing attack vectors, a typical purpose of numerous vectors is the utilization of link misleading victims to phishing websites, this fact motivates us to build wide scope detection system by employing URLs features only.

A lot of works have been directed by researchers [3,4,5,6,7,8] with the end goal of forestall or decrease a phishing attacks and numerous thoughts have been proposed in this field. The early methodology was depending on blacklists, which is a basic and not productive because of the reliance on the remote database to look at the website as phishing or benevolent. On the other hand, intelligent solutions are more productive in comparison with blacklists, intelligent solutions depend on extracting important features of the website and after the extraction process these features utilized by an algorithm to decide or detect the phishing website.

Intelligent solutions in general can be divided into content based and URL based solutions. Content based intercept and download the full contents of website for analyzing which can provide high detection accuracy with much more runtime overhead. In addition, it might accidentally provide more threats to users they look to keep safe from it. URL based techniques use a combination of host information and lexical features [9, 10]. Hosting information needs external server beside the huge feature vector generated by a bag of word method poses high latency preventing the application of such method for real time. However, URL based methods proved to be a good option to fight phishing attacks [11]. Mostly, URL features are used to train a machine learning algorithms (ML) to generate a classifier to detect unseen URLs.

Internet users are usually very impatient if they got a delay in obtaining information so that, the development of any detection system must be fast enough to not annoy users surfing of Internet. However, nowadays the highest accuracy system the highest time consumption. Although that previous studies employed the URLs lexical features, no one tries to build phishing detection system using a pure lexical features to provide wide scope, lightweight, and highly accurate classifier.

2. RELATED WORK

Content based phishing detection systems load the website first to identify whether the websites is phishing or legitimate which ultimately expose Internet users to phishing conducted using malicious codes. Therefore, to overcome this danger, Garera [12] proposed a phishing prevention approach that uses only anomalies in the URLs of phishing websites to detect them. As users deal with URLs directly to get contents from Internet, many obfuscation methods used by phishers to generate trustworthy looking URLs.

Authors in [13] synthesis the characters frequency of phishing URLs. Their findings show that phishing domains usually use different characters and vowels besides inserting the name of the target brand. Also, they state that short domain names and long URL are good signs for phishing. Therefore, numerous techniques are proposed based on URLs lexical features [14,15,9,16]. Such lexical features are the length of URL, number of dots, number of domains, and etc. One of the most important strengths of this method, the extraction of such features is not time consume and prevent the danger and latency of the page loading. In general, URLs features are used with machine learning techniques to build the classifiers to detect the unseen input samples.

The work in [17] is one of the earlier attempts applied machine learning to detect phishing URLs using lexical features with the bag of words representation. It shows that the method can provide accuracy of 95%. This high accuracy also confirmed by the results in [18] that machine learning with lexical features can provide such high accuracies.

PhishStorm proposed by Marchal [19] is an automated phishing detection system which based on analysis of URL lexical in real time environment. The system implemented as central detection unit places in front of the email server. 12 features of URL extracted using the searching engines followed by supervised classification step. The PhishStorm provided 94.91%, 1.44% accuracy and false positive respectively. However, this method is time consumed due to the searching engines.

Lastly, some methods combine lexical features with other features type such as contents of website and host information. Such work is noted in [10] which implement a real time system to classify the URLs. The accuracy provided by this work about 91% with 5.54 seconds required to process a single URL. This amount of time bothers the users when they surfing the internet and make it not good option for real time application.

In the view of the related work, URL based techniques can provide high accuracies over 90% and required little information without need for more contents. In addition, it can be used to cover a wide scope of phishing attacks vector. However, the time required for URL processing still subject of research besides most of the work does not report the timing analysis and the time provided in [10] is not suitable for large scale systems.

3. RESEARCH METHODOLOGY

The methodology consists of three main phases where the output of each phase being next phase input. The first phase is the collection of datasets from different sources. Phase two comprises datasets preprocessing, building a language module, and feature extraction. The last phase is classifiers evaluation based on evaluation metrics. In addition, the results from each classifier to be compared in this phase to highlight the best technique based on the performance metrics used in this work.

3.1 Phase 1: Dataset collection

We collected 46,5461 phish websites from Phishtank which have been accumulated over a period since 2008 January. We propose a novel method to divide Phishtank dataset into three different datasets according to the years they appear in the dataset. These datasets namely D2013, D2014, and D2015. For more closely following the evolving features of phishing URLs and to mimic the real-world scenario, we collected a second batch of 4647 confirmed phishing URLs that were submitted to OpenPhish. Since it launched recently, none of the literature that we studied had access to this data for pure phishing detection.

To cover the diversity of legitimate websites, our legitimate URLs are gathered from two data sources provided publicly: DMOZ.org and webcrawler.com. We use 10,275 randomly chosen non-phishing URLs from DMOZ and we call it DMOZ data set. Also, we collected 10,275 URLs from webcrawler and we named this dataset as WebCrawler.

We paired PhishTank and Openphish data sets with non-phishing URLs from a benign source (either DMOZ or WebCrawler). We refer to these data sets as the D2013-DMOZ (D13DM), D2014-DMOZ (D14DM), D2015-DMOZ (D15DM), D2013- WebCrawler (D13WC), D2014-WebCrawler (D14WC), D2015- WebCrawler (D15WC), OpenPhish-DMOZ (ODM) and OpenPhish-WebCrawler (OWC). Figure 1 depicts the methodology of dataset dividing and merging.

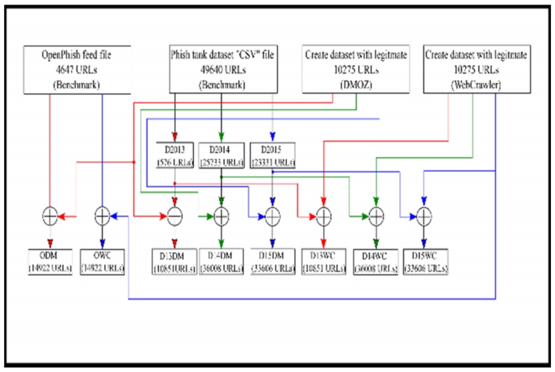


Figure 1. Dataset Division And Merging Methodology.

3.2 Phase 2: Dataset Processing, N-Gram Model, And Feature Extraction

Data preprocessing is to get an understandable format from the collected datasets (raw data) because of inconsistently, incompleteness, and certain behaviors lacking among the main features of real life data. To fix such issues, data preprocessing is implemented to make the raw data format suitable for further processing. The dataset from Phishtank contains a number of basic columns, for instance: phishing verified, brand name target, and the URL of the phishing website. Openphish dataset contains more basic columns such as sector, IP, brand, isotime, host, country code, etc. At the end of this step, we will remove unnecessary columns and combine the similar columns. After getting the combined dataset it converted into MySQL in order to assist us for the next steps.

Assigning probability for strings of any language is the primary goal of the language model. The better language model is the one will produce low probabilities for ungrammatical and uncommon strings meanwhile for the grammatical and common strings assign high probabilities. Building on this concept we tried to construct a model of the language of URLs.

Parsing N-sized grams from the training dataset is a general method to generate the language models where N is an integer. Markov chains are used to construct the N-grams where this method based on the probability of a gram with the assumption that the probability of any gram just depends on the previous gram probability.

We calculate the probability of characters sequence in URLs using the N-gram model whereby a lot of works proved that the N-gram method is the best option to generate the language model and tough to improve on it [20].

In spite of the N-gram method to build the URL language model is implemented in previous work [17] and used for malicious websites detection [21], to our knowledge this technique is not used for pure phishing URL detection yet. This work tries to implement unigram, bigrams, trigrams and fourgrams. We proposed methodology to create our N-gram models based on benign dataset only.

To represent our method mathematically, we assume to map the legitimate URLs into N sized sequence of strings (N-gram) e.g. (W_1……W_k) where N is an integer numbers from 1 to 4 and k is the number of N sized strings exist in the legitimate datasets. In order to compute each gram probability, Maximum Likelihood Estimation (MLE) is employed by calculating the frequency of each gram in dataset. In this way, computing the probability of any unigram is achieved by dividing the occurrences number of that unigram by the sum of all other unigrams occurrences count as showed in (1). To calculate any bigram probability, the bigram frequency is divided by the count number of the first half string of that bigram. In the same manner, the trigram and fourgram probabilities are computed using the general formula as shown in (2).

$$p \qquad (w) = \frac{C(w)}{\sum_{i=1}^{k} C(w)} \tag{1}$$

K is the number of dataset' unigrams.

$$p \qquad (w_n | w_{n-N+1}^{n-1}) = \frac{c(w_{n-N+1}^{n-1}\, w_n)}{c(w_{n-N+1}^{n-1})} \tag{2}$$

Where N is size of gram takes value from two to four and n represents the number of strings that should be looked in the past.

3.3 Phase 3: Individual Classifier Evaluation

Classification is predicting the class label of input samples. There are two outputs in a binary classification problem such in phishing detection in this paper the output is either "1" or "0". Several methods can be used to measure classification performance where the most popular metrics are accuracy, False Positive Rate (FPR), False

Journal of Theoretical and Applied Information Technology
30th June 2016. Vol.88. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Negative Rate (FNR), True Positive rate (TPR), True Negative rate (TNR), and error rate. Accuracy is not enough to clearly evaluate the classifiers performance due to there is no distinction between classes where correct answers of class 1 and class 0 are equally treated. So that, the rest metrics present more information about the incorrect and correct decisions for each class. In real world, the error rate of any class can be at a higher cost than wrongly identifying other class. Also, the time required to process a single URL is a very important factor to test the system suitability for working in a real time environment, the minimum processing time always the better. At the end of this phase, the best classifier is selected based on the highest performance.

- **False Positive Rate (FPR)**: Defined as the ratio of legitimate class that incorrectly classified as a phishing to the total number of legitimate class instances.

$$F = \frac{N \to p}{N \to l + N \to p} \qquad (3)$$

- **False Negative Rate (FNR)**: Defined as the ratio of phish class that incorrectly classified as legitimate class to the total number of phish class instances.

$$F = \frac{N \to l}{N \to p + N \to l} \qquad (4)$$

- **Recall or True Positive rate (TPR)**: Defined as the frequency of patterns which are detected correctly by the classifier as a phish.

$$T = \frac{N \to p}{N \to p + N \to l} \qquad (5)$$

- **True Negative rate (TNR)**: Defined as the frequency of patterns which are detected correctly by the classifier as a legitimate.

$$T = \frac{N \to l}{N \to l + N \to p} \qquad (6)$$

- **Accuracy**: Defined as the percentage of correct classification over all attempts of classification.

$$A = (N \to p + N \to l)/(N \to p + N \to p + N \to l + N \to l) \qquad (7)$$

- **Classification Time**: Defined as the total time of feature extraction and classifying the input instance.

$$Ti = \text{Features extraction time} + T \quad ti \qquad (8)$$

4. IMPLEMENTATION AND RESULTS

The n-gram model has been constructed separately for the URLs host, path, and query using the legitimate URLs only. A Java codes are developed to read these datasets and calculate the probability of each gram occurs in the datasets. The flowchart in Figure 2 illustrates the implementation flow of n-gram model software.

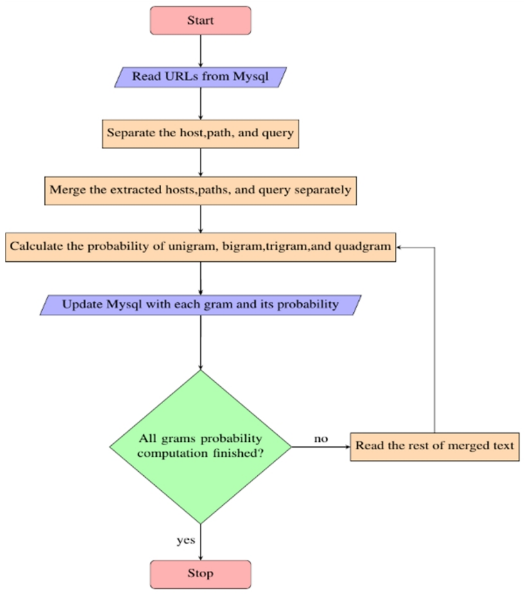

Start
Read URLs from Mysql
Separate the host,path, and query
Merge the extracted hosts,paths, and query separately
Calculate the probability of unigram, bigram,trigram,and quadgram
Update Mysql with each gram and its probability
All grams probability computation finished?
no
Read the rest of merged text
yes
Stop

Figure 2: N-Gram Model Building Flowchart.

To extract the 12 N-gram features, MySQL databases have been developed for features extraction. Table 1 shows the columns of a MySQL table and describe them briefly.

To extract features from URLs, the Java code reads the URL and splits it into host, path, and query and extract all grams exist in each part and kept as a list of text. After this step, the grams of each part are read in sequence and MySQL database is looked up against each gram to find its probability. If the gram probability not exists, zero probability is simply assigned. The extracted grams probability of the host, path, and query is added separately to generate the value of each feature.

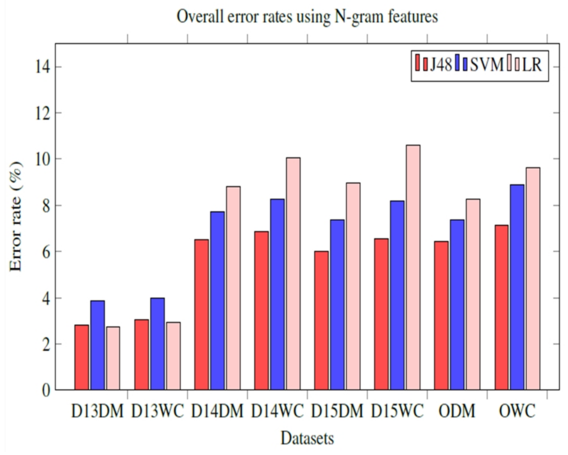
After all the required features have been fetched successfully, the features vector needs to be converted into ARFF format to be used as input to the classifiers. In this work, we used three classifiers

Table 1: Mysql Database Columns With Description.

N.	Name	Column Description
1	ID	Unique number of each row
2	URLs	URL of the websites which are imported from Phishing and legitimate sources.
3	UniGram_Host	The probability summation of the unigrams extracted from host n-gram model.
4	BiGram_Host	The probability summation of the bigrams extracted from host n-gram model.
5	TriGram_Host	The probability summation of the trigrams extracted from host n-gram model.
6	FourGram_Host	The probability summation of the fourgrams extracted from host n-gram model.
7	UniGram_Path	The probability summation of the unigrams extracted from path n-gram model.
8	BiGram_Path	The probability summation of the bigrams extracted from path n-gram model.
9	TriGram_Path	The probability summation of the trigrams extracted from path n-gram model.
10	FourGram_Path	The probability summation of the fourgrams extracted from path n-gram model.
11	UniGram_Query	The probability summation of the unigrams extracted from query n-gram model.
12	BiGram_Query	The probability summation of the bigrams extracted from query n-gram model.
13	TriGram_Query	The probability summation of the trigrams extracted from query n-gram model.
14	FourGram_Query	The probability summation of the fourgrams extracted from query n-gram model.
15	Label	The classification of each webpage, 1 mean phishing and 0 mean legitimate.

which are J4.8, SVM, and LR implemented in widely known data mining tools in the research community the Waikato Environment for Knowledge Analysis (WEKA) [22]. All classifiers evaluated using its default parameter values. The 10-fold cross-validation is used to evaluate the classifiers performance on each indivisible dataset. The machine with 8 GB memory and core-i7 2.57 GHz processor is used to run all the experiments.

The extracted N-gram features are used to evaluate the performance of three classifiers on all indivisible datasets. Figure 3 compares the results of the classifiers: J48, SVM, and LR.



Classifiers error rates on each of the data sets are shown by the bars. Classifiers error rate range from

Figure 3: Classifiers Error Rates.

2.7% to 10.6%. Clearly, J48 provides the lowest error rate range between 2.8% to 7.1% on all dataset. SVM comes after J48 in error range from 3.8% to 8.9%. LR ranked last with an error rate of 2.7% to 10.6%.

In order to take a closer look and to analyze in depth, Tables 2 and Table 3 show that the differences in overall accuracy rates on all classifiers are not significant on D13DM and D13WC datasets with range from 96% to 97.28%. LR performs best in TPR (51%-56.4%). Although, all classifiers perform very well in detecting the legitimate URLs with 100% TNR offered by SVM on D13DM. However, classifiers yield the worst performance among all datasets in term of TPR because of the high FNR which is a results of unbalance datasets. It is clear that all classifiers are not trained with adequate samples of phishing to recognize the phishing URLs from the legitimate URLs.

TABLE 2: Classifiers Performance On D13DM.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.533	0.004	0.996	0.467	97.16%	2.83%
SVM	0.269	0	1	0.731	96.11%	3.88%
LR	0.564	0.004	0.996	0.436	97.27%	2.72%

TABLE 3: Classifiers Performance On D13WC.

	TPR	FPR	TNR	FNR	Accuracy	Error

J48	0.474	0.003	0.997	0.526	96.93%	3.06%
SVM	0.26	0.001	0.999	0.74	96.01%	3.98%
LR	0.51	0.004	0.996	0.49	97.06%	2.93%

More phishing samples the more TPR this is proved by the results shown in Table 4, Table 5, Table 6, and Table 7. All classifiers are trained with sufficient number of phishing samples which in turn give classifiers the ability to distinguish phishing instances. The highest TPR of 97.1% achieved by J4.8 with 93.97% accuracy. In spite of increasing phishing samples give such increase in TPR, overall classifiers accuracies decreased due to the increase of FPR with the highest value reached up to 21% by LR. In general, J4.8 is the best classifier followed by SVM and LR respectively.

TABLE 4: Classifiers Performance On D14DM.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.971	0.156	0.844	0.029	93.50%	6.49%
SVM	0.959	0.168	0.832	0.041	92.27%	7.72%
LR	0.948	0.179	0.821	0.052	91.17%	8.82%

TABLE 5: Classifiers Performance On D14WC.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.969	0.163	0.837	0.031	93.15%	6.84%
SVM	0.959	0.186	0.814	0.041	91.74%	8.25%
LR	0.943	0.21	0.79	0.057	89.92%	10.07%

TABLE 6: Classifiers Performance On D15DM.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.967	0.122	0.878	0.033	93.97%	6.02%
SVM	0.956	0.141	0.859	0.044	92.63%	7.36%
LR	0.934	0.144	0.856	0.066	91.04%	8.95%

TABLE 7: Classifiers Performance On D15WC.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.963	0.13	0.87	0.037	93.44%	6.55%
SVM	0.955	0.167	0.833	0.045	91.80%	8.19%
LR	0.93	0.188	0.812	0.07	89.39%	10.60%

Table 8 and Table 9 present the classifiers performance on ODM and OWC, which confirm the results of D14DM, D14WC, D15DM and D15WC. J4.8 provides the best accuracy with value 93.55% with good balance in FPR and FNR.

The observation from the experimental results, to increase classifiers performance it is very necessary to use balanced dataset in the training stage. In comparison with other bag of word methods which may generate handers or thousands of features, just 12 N-gram features give good technique to detect phishing URLs as proved by the explained results. Generally, the justification for the success of this method that as N-gram model builds using legitimate URLs, the system can differentiate phishing URLs because of the phishing grams will not characterize in the models of N-gram.

TABLE 8: Classifiers Performance On ODM.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.916	0.055	0.945	0.084	93.55%	6.44%
SVM	0.889	0.057	0.943	0.111	92.62%	7.37%
LR	0.817	0.037	0.963	0.183	91.75%	8.24%

TABLE 9: Classifiers Performance On OWC.

	TPR	FPR	TNR	FNR	Accuracy	Error
J48	0.901	0.059	0.941	0.099	92.84%	7.15%
SVM	0.884	0.077	0.923	0.116	91.09%	8.90%
LR	0.782	0.041	0.959	0.218	90.39%	9.61%

5. TIME OF TRAINING AND TESTING

Bearing in mind the end goal to keep Internet users safe from visiting phishing websites in real time, the classifier should be very accurate besides producing minimal processing time. We explore the possibility of the application of the J48 classifier in real time. In this test, the time consumed by the classifier to detect single URL as phishing or not is investigated by computing the time taken from extracting the features to test and providing the final results.

Experimentally, the proposed classification system can generate the features vector of a single URL with format discernable by the classifier in average time of about 0.12 second. The justification for this superior processing time due to our system based on URL lexical features only and there are no needs for external features.

To measure the time of training and testing, the D14DM is split in rate of 80/20 in order to give better examinations of the time consumed by the classifier to build and test the generated module. Also, the required time to classify a single URL is presented. Table 10 shows the average results after 3 times experiment running.

In spite of the time required to generate the model higher than the testing time, the process of building the classifier is less frequently and can be seen as an offline process. The time needed for features extraction and testing is the decisive factor to provide real time detection. Once the classifier generated, the time consumed to identify a single URL is very low and negligible in comparison with feature collection time. As a result, the proposed classifier in this work can classify single URL in average time of 0.12 second.

TABLE 10: J48 Training And Testing Time.

Samples number	Average time
28806	Train=0.848 second
7202	Test=0.021 second
1	Test=0.001 second

Though the datasets different in somehow, this work can be compared with results in [10, 16, 19] where their methods based on features extracted from URL only. From results, the average accuracy provided by J48 about 94%. Unlike previous studies, this accuracy level is achieved without tedious lookups as shown in Table 11. Again, this superior processing time is achieved due to the proposed classifier utilizes URL lexical features only.

TABLE 11: Processing Time And Accuracy Comparison With Other Works.

Author	year	Accuracy	Processing time (second)
Thomas	2011	91%	5.54
Basnet	2014	99.4%	3.5
Samuel	2014	94.91%	0.77
Our method		94%	0.12

Despite the encouraging results obtained from N-gram features, this method is not without its shortcomings. Our method can be effected if the attackers use the shortening service due to the system extracts features from the full length of URLs. Shortening service is becoming widely used by the general public. Clicking on a shortened URL will redirect the user to the webpage of the full length URL. This feature is exploited by phishers to fraud Internet users by hiding the original URL and to bypass the current detection systems.

6. CONCLUSION AND FUTURE WORK

New lightweight and wide scope phishing detection classifier is presented in this paper. The proposed classifier provides accuracy 94% on average and 0.12 second to classify single URL. This accuracy level is achieved without huge time consuming as in previous works. Although, N-gram based features provide high accuracies, there still a gap to further improve the accuracies and reduce the error rate. We believe adding the most effective and lightweight bag of word features will improve the accuracy and reduce the error rates rapidly.

REFRENCES:

[1] Khonji, M., Iraqi, Y., & Jones, A. (2013). Phishing detection: a literature survey. *Communications Surveys & Tutorials, IEEE*, *15*(4), 2091-2121.

[2] Liu, G., Qiu, B., & Wenyin, L. (2010, August). Automatic detection of phishing target from phishing webpage. In *Pattern Recognition (ICPR), 2010 20th International Conference on* (pp. 153-4156). IEEE.

[3] Abdelhamid, N. (2015). Multi-label rules for phishing classification. *Applied Computing and Informatics*, *11*(1), 29-46.

[4] Afroz, S., & Greenstadt, R. (2011, September). Phishzoo: Detecting phishing websites by looking at them. In *Semantic Computing (ICSC), 2011 Fifth IEEE International Conference on* (pp. 368-375). IEEE.

[5] Hamid, I. R. A., & Abawajy, J. H. (2013, July). Profiling phishing email based on clustering approach. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*(pp. 628-635). IEEE.

[6] Khonji, M., Iraqi, Y., & Jones, A. (2012). Enhancing phishing E-Mail classifiers: a lexical URL analysis approach. *International Journal for Information Security Research (IJISR)*, *2*(1/2).

[7] Lee, L. H., Lee, K. C., Chen, H. H., & Tseng, Y. H. (2014, November). POSTER: Proactive Blacklist Update for Anti-Phishing. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1448-1450). ACM.

[8] Liu, G., Qiu, B., & Wenyin, L. (2010, August). Automatic detection of phishing target from phishing webpage. In *Pattern Recognition (ICPR), 2010 20th International Conference on* (pp. 4153-4156). IEEE.

[9] Le, A., Markopoulou, A., & Faloutsos, M. (2011, April). Phishdef: Url names say it all. In *INFOCOM, 2011 Proceedings IEEE* (pp. 191-195). IEEE.

[10] Thomas, K., Grier, C., Ma, J., Paxson, V., & Song, D. (2011, May). Design and evaluation of a real-time url spam filtering service. In *Security and Privacy (SP), 2011 IEEE Symposium on* (pp. 447-462). IEEE.

[11] Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009, June). Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In*Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1245-1254). ACM.

[12] Garera, S., Provos, N., Chew, M., & Rubin, A. D. (2007, November). A framework for detection and measurement of phishing attacks. In*Proceedings of the 2007 ACM workshop on Recurring malcode* (pp. 1-8). ACM.

[13] McGrath, D. K., & Gupta, M. (2008). Behind Phishing: An Examination of Phisher Modi Operandi. *LEET*, *8*, 4.

[14] Blum, A., Wardman, B., Solorio, T., & Warner, G. (2010, October). Lexical feature based phishing URL detection using online learning. In *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security* (pp. 54-60). ACM.

[15] Khonji, M., Iraqi, Y., & Jones, A. (2011, September). Lexical URL analysis for discriminating phishing and legitimate websites. In *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference* (pp. 109-115). ACM.

[16] Basnet, R. B., Sung, A. H., & Liu, Q. (2014). Learning to detect phishing URLs. *IJRET: International Journal of Research in Engineering and Technology*, *3*(6), 11-24.

[17] Kan, M. Y., & Thi, H. O. N. (2005, October). Fast webpage classification using URL features. In *Proceedings of the 14th ACM international conference on Information and knowledge management* (pp. 325-326). ACM.

[18] Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2011). Learning to detect malicious urls. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 30.

[19] Marchal, S., François, J., State, R., & Engel, T. (2014). PhishStorm: Detecting Phishing With Streaming Analytics. *Network and Service Management, IEEE Transactions on*, *11*(4), 458-471.

[20] Jelinek, F. (1991). Up from trigrams. Eurospeech.

[21] Darling, M., Heileman, G., Gressel, G., Ashok, A., & Poornachandran, P. (2015, July). A lexical approach for classifying malicious URLs. In *High Performance Computing & Simulation (HPCS), 2015 International Conference on* (pp. 195-202). IEEE.

[22] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, *11*(1), 10-18.