RPhish: A Multi-modular Machine Learning Based Tool for Phishing Website Detection Focusing on Faster Prediction Time

Risul Islam
University of California, Riverside
risla002@ucr.edu

Md. Omar Faruk Rokon
University of California, Riverside
mroko001@ucr.edu

Michalis Faloutsos
University of California, Riverside
michalis@cs.ucr.edu

ABSTRACT

How can we detect whether a website is phishing or legitimate? This question tends to be a very important one as many of the internet users suffer from Phishing attack. Phishing is considered a form of web-threats that is defined as the process of impersonating an original website aiming to acquire private information such as usernames, password's and social security numbers. So, internet-users may be vulnerable to different types of web-threats that may cause financial damages, identity theft, loss of private information, brand reputation damage and loss of customer's confidence in e-commerce and online banking. Phishing attack is continuously taking new forms with the passage of time as the contents/features in the website are continuously changing. In this paper, we propose RPhish, a multi-modular machine learning based tool to predict phishing website in faster real time with good accuracy. So, we envision Rphish as having many modules, each working with a subset of features collected from a phishing dataset of with 30 features , with an aim to detect phishing websites with different characteristics. We have run forensics on a lump sum of URL Typosquatting of some real websites and 100 phishing websites to have a real view of their characteristics. However, different modules of RPhish will use different prediction algorithms like KNN, SVM, DNN, Anytime KNN etc. Many of these algorithms/architectures are novel in this phishing website detection domain- some showing good results, some not. So, we have tasted not only the best outcomes but also the ways in which we can be failed. For having this multi-modular concept, RPhish is going to be very robust, scalable, faster and accurate tool to come in great help for the internet end-users.

CCS CONCEPTS

• **Security and privacy** → **Phishing**; Use https://dl.acm.org/ccs.cfm to generate actual concepts section for your paper; • **Information systems** → Collaborative filtering; Deep web;

KEYWORDS

Phishing; Typosquatting; Feature Compression; Neural Network; Prediction time; Accuracy

1 INTRODUCTION

Phishing attack actually targets the internet users rather than the computer system unlike virus or malware. So, every internet user is a probable victim for the attacker. Phishing attack is performed when a user visits an identical looking fake website rather than the original website. When the user falls in this trap he is then lured to give his sensitive information. That is how the attacker can get the information by what he can utilize the personal information for bad purpose.

Phishing attack is becoming a massive problem as many of the financial and social sites are being targeted. Though the companies are reluctant to show the exact figures of the damage caused by the attack, estimation shows about 1 billion [14] to 2.8 billions [15] per year loss. Still people are falling into this pit. In a study, it has been shown that a large number of people can not differentiate between legitimate and phishing site even if they are made aware of that their ability is being tested [11]. To fight against this phishing attack, many social as well as technical approaches have been taken. Online training materials by government [8, 9], non profit[4] and business organizations [13, 16] have been published. Mail Frontier has an website containing the screenshots of the Phishing websites. Along with these social approaches, legal bindings and laws have been imposed. Phishing had been added as a computer crime in the list first by USA on January 2004 by FTC and in March 2005, the 'Anti-Phishing Act' was first introduced in Congress by senator 'Patrick Leahy' under which many have been arrested and sued. Still, the number of attack is not decreasing. Researchers have then come out in this regard and introduced many technical solutions. There are actually two types of solution. 1. URL Filtering: detects phishing sites by comparing the URL from the address bar with a Blacklisted Phishing URLs. But this approach tends to be not working as the number of phishing sites is growing very fast. 2. Heuristics Based Solution: extracts some features based on some heuristics of that web page and analyzing the features it tells whether the website is a phishing. For analyzing the features, Machine Learning Techniques tend to be very useful. In this technique, a set of features is used to build a model. Then using that model we classify a website as Phishing or Legitimate. So, it is a classic binary classification problem.

In our work, we have used 30 features and divided them in three categories eg. name based category (Category A), network/ reputation based category (Category B) and content based category (Category C). Then from different combinations of these categories and pre-collected dataset, we have built models and thus predict the website as phishing or not using 9 different machine learning algorithms. For example, we have used 8 features from the name based category (Category A) from our dataset and then used KNN for prediction- we call this system a 'Module'. So, in Module A, we use features from name based category (Category A) and so on. Another example is that, in Module AB, we have used features from name based and content based features from our dataset and used Random Forest algorithm for classification. In this manner, we can take 30 features from all three categories (Module ABC) for the prediction to get the best accuracy and trusted result. In this way we have built actually multi-modular phishing detection. But the problem is that when we are using all 30 features from all three categories, it takes longer prediction time causing the users to wait.

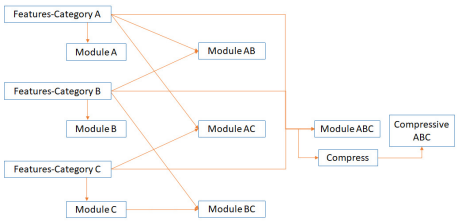


Figure 1: A brief demonstration of our Multi-modular and compression based data mining techniques for phishing detection

To remove this time constraint, we have used another technique called 'Feature Compression'. We have used three compression techniques like PCA, Gaussian Random Projection, Sparse Random Projection on the features (only when we are using all 30 features) and then build the model for prediction which takes reasonably less prediction time (PCA works the best). The basic block diagram is demonstrated in Figure 1. We have also used some inter-category features to detect 'click farms' as well as phishing. To improve accuracy and prediction time without overfitting, we have introduced 'Module Deep' which actually uses Neural Network Architecture. And lastly we have added options like 'Module Majority Vote' which actually uses different Modules' predictions and take the majority vote, 'Module myFeature' to give the users a opportunity to set their own feature set and prediction algorithms, 'Module Anytime' to interrupt the prediction computation if it takes longer time and have a quick result. All these Modules help to detect all kinds of Phishing sites as the sites are being very smart now-a-days and do not exhibit only some special kinds of features. So multi-modular approach of RPhish is very robust (able to detect different variations of Phishing sites) with trusted best accuracy and prediction time. We have got the motivation for this project from a comprehensive forensics on real websites (Phishing and legitimate). We will now point out our contributions briefly.

- A comprehensive forensics on 3759 websites of eight categories with a special focus on the URL Typosquattings of chase.com and 100 real phishing websites reported in PhishTank.com
- multi-modular system, each module uses features from a particular subset from a feature set of 30.
- introduced novel ideas like 'Feature Compression' and 'Feature Selection', Anytime algorithms on this 'Phishing Website Detection' domain to have better accuracy with less prediction time so far.
- applied different Neural Network Architectures (almost all of them are novel in this domain) to gain least prediction time with best accuracy.

The rest of the paper is organized as follows. Section 2 states the Problem Definition and section 3 describes the related works . Section 4, 5 and 6 describes the Forensics, our whole approach and Evaluation respectively. And lastly, section 7,8 and 9 describes the future works, conclusion and reference respectively.

2 PROBLEM DEFINITION

The problem we address here is how we can detect whether a website is phishing or not. So, the input of our problem is a website page with URL and the output is that we will tell whether it is a phishing site or legitimate site. Our challenges here are that we will have to detect it in real time with good accuracy. We will use some conventional machine learning algorithms and particularly focus on how we can predict whether websites, with varying properties/characteristics, are phishing or not before Google blacklists those and decrease the prediction time, storage and computation complexity with reasonable accuracy.

3 RELATED WORKS

Our work is novel in the sense that we are using various compression algorithms and Neural Network Architectures to lower the real time prediction time with reasonable accuracy, TPR and FPR. We will compare them with Daisuke Miyamoto et al. work [17] who had used 8 heuristics (features) and 9 machine learning algorithms to get 88% accuracy, Yue Zhang et al. work [27] who used TF-IDF algorithm with too few heuristics (7) to get 95% accuracy, Maher Aburrous et al. [3] work who used 27 heuristics to get 84% accuracy and P.A. Barraclough et al. work [6] who uses neuro-fuzzy network with too high number of features (287) to get 93% accuracy with the cost of too long prediction time. Our work uses 30 features and gets accuracy maximum 98.21%, reasonably lower FPR, Higher TPR and faster prediction time.

Though there are a lot of existing phishing detection tools and techniques, they are not giving that accurate results as they are showing high amount of false positive. In this section, we will discuss some recent contribution in phishing detection techniques. In [1], the authors utilized several machine learning methods including LR, CART, RF, NB, SVM, and BART to measure the accuracy to detect phishing 'email'. They used 43 features to analyze 1,117 phishing emails and 1,718 legitimate emails. Random Forests showed the lowest error rate, 7.72%. In [7] , the authors evaluated six different machine learning-based detection methods to analyze 973 phishing emails and 3,027 legitimate emails with 12 features, and showed that the lowest error rate was 2.01%. The experimental conditions were different between [1] and [7], however, the machine learning provided high accuracy for the detection of phishing emails.

In [3], the authors categorized 27 features and applied Classification Based Association (CBA), and Multi-class Classification based on Association Rule (MCAR) with other classification algorithms using data mining techniques. They identified a significant relationship between 'URL' and 'Domain-Identity' features. After that, in 2010 [2], they built a promising solution, a fuzzy-logic based model using 27 features. However, it did not provide details how they extracted the human factors related features from the websites. Besides that, it used human interaction whereas our main concern is to keep the process completely human interaction free.

In [18], Mohammad at el. proposed a method to extract phishing features of a website. They identified these features as Dataset preparation, Address bar features, Abnormal based features, HTML and JavaScript based features, and Domain based features. In [20], the authors proposed an approach to detect phishing sites using six abnormal behavior of these sites. Abnormal URL, Abnormal Cookie,

Abnormal DNS record, Abnormal SSL, Abnormal Anchors, and Server Form Handler (SFH) are the six structural behavior which were used in the SVM classfier Vapinik's [10] in order to detect the phishing sites. Though it showed only 84% accuracy, it played an important role to design features in phishing site detection.

In another method, proposed in [27], the authors suggested a content based technique, known as CANTINA, to detect phishing site using TF-IDF which gives weight of the content in a webpage counting frequency of every word. The top five TF-IDF terms is used to get the lexical-signature which is then fed to a search engine to measure the abnormality of the site. However, as it gave high false positive rate, they combined some other features such as Suspicious URL, IP Address in URL, Age of Domain, Known Images, Dots in URL and Forms with TF-IDF to get better results. Another approach in [23], the authors used different machine learning techniques utilizing CANTINA along with another attribute. They showed that additional features in CANTINA had increased the accuracy in detecting phishing site. Among different machine-learning-based algorithms, Neural network gave the best result with only 7.50% error-rate and Naive Bayes showed the worst with 22.5% error rate. Xiang et al. [26] proposed CANTINA+ where they used eight features. They utilizes HTML DOM, search engines and third party services with machine learning techniques for detecting phishing websites. However, both CANTINA and CANTINA+ can not detect image phishing as they deal with only texts.

In [19], Mohammad at el. used self-structuring neural networks to predict phishing website. Back-propagation algorithm has been used in the networks. They have used 17 features of the phishing websites as input to the neural networks. However, the final network architecture is very simple with only one hidden which may end up with bad accuracy in this current era of evolving phishing website and prediction time was not their primary concern. The prime difference between their and our architecture is that we have introduced two hidden layers instead of three and used 30 features. As a result, our test set and validation accuracy, epochs, error results are way better than them.

Meanwhile some researchers proposed feature selection as an intuitive approach to reduce the dimensionality of the feature space and improve the accuracy. In [22], the authors established three characteristics Information Gain, Chi Square and CFS on 47 features for phishing 'email' filtering. However, no researcher has used feature selection in phishing web site detection. We have used forward selection, backward elimination and modified correlation based filter method for feature selection.

However, the core idea of our system is to compress the features to improve the prediction time, computational and storage complexity- thus improving the scalability. Note that an algorithm A is more scalable than B if (i) both offers same/reasonable accuracy but A has improvement in storage, computation and time complexity (ii) both offers same storage, computation and time complexity but A shows higher accuracy. In a sense our work follows condition (i) and actually shows better accuracy and resource managemant. Our work is orthogonal from the previous work in the sense that we have introduced a vast forensics, some novel machine learning algorithms (some of them showed good results, some did not) in this domain with a focus on prediction time with reasonable/best accuracy seen so far. One thing to mention that recently there are a limited works going on phishing website detection. Probably it is due to 95% accuracy gained by CANTINA and there was a huge chance of overfitting for their dataset. But our works show that RPhish can have maximum 98.21% accuracy without overfitting. As for having many modules incorporating the previous works and new works, RPhish can fill the timehole of phishing website detection research and direct the future researchers.

4 FORENSICS

4.1 Finding and analyzing URL Typosquatting to find Phishing website

Clever phishing attacker buys domains which names are close to original website's name. Depending on this observation, we have analyzed some website's close domain. Note that these phony close domains that are common misspelling of original websites are called 'URL Typosquatting' . So, we have first chosen eight categories and from each category we have chosen one website. We have generated the possible Typosquatts totaling to 3759 by replacing one character in the original websites name (1-mod-inplace), by inserting one additional character (1-mod-infaltion) and deleting one character (1-mod-deflation) following the works of Anirban et.al. [5]. We have found out how many of them are now 'existing' in the web and how many of the 'existing' website is Google indexed. From this observation we have found out that banking and social sites are mostly prone to URL Typosquatting. Then we mainly focused on one banking website (www.chase.com) for further deep investigation. Note that we have only generated the possible Typosquatts with '.com' extension. We have analyzed the possible misspelling of chase.com existing in the web using our own algorithm. Though we have found a lower number of phishing website using this approach, we have found out some important findings which have helped us to form a module based feature analyzing to detect phishing website. Table 1 and 2 will summarize the findings.

Note that there are 390 possible Typosquatts for chase.com and we have found out 186 existing in the web. We will now point out the findings from our analysis and obviously we have analyzed only the existing websites. The summary is given below:

- 33.8% of the Typoquatts has HTTPS. Out of this 33.8%, 25% has invalid HTTPS; their certification for HTTPS have been invalidated.
- 11.2% shows Phishy URL after redirection (if redirected). We mean by Phishy URL that the URL is too big or too small or having '@' symbol or having '-' symbol etc.
- 9% shows pop-up and 3% wanted sensitive information in pop-up. Surely these are Phishing website.
- 54.8% is redirected and out of it 60% is redirected to HTTPS site and 40% is redirected to non-HTTPS site. This non-HTTPS redirections are prone to suspicious or Phishing. 40% of the sites is redirected by more than 3 times.

4.2 Malicious, Legitimate or Suspicious observation

Observation 1:

Table 1: Summary result of analysis of URL Typosquatting of 8 prominent websites

Category	Name	'Existing' site out of all Typosquatt (%)	Not Existing(%)	Google Indesxd in 'Existing' sites(%)
Banking	Chase	66	34	36
Online Shopping	Amazon	20	80	41
Search Engine	Google	53	47	26
Adult Site	Pornhub	52	47	26
Interactive	Reddit	46	54	19
Social Networks	Facebook	69	31	27
Video Sharing	YouTube	60	40	31
News Portal	Nytimes	35	65	30

Table 2: Special Characteristics of URL Typosquatting of 'Chase' Bank

Characteristics	Values(%)
Google Indexed in Existing sites	36
Suspicious in Existing	22.6
Suspicious in Google Indexed	25.75
Phishing in Existing	2.15

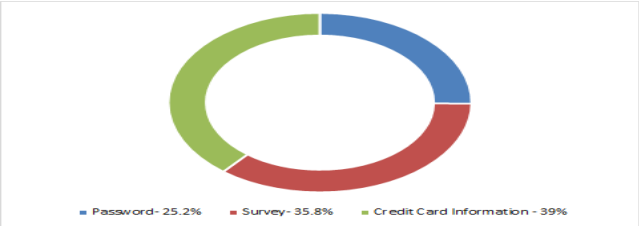


Figure 2: Forensic results: Phishing action performed by Phishing website in URL Typoquatting of Chase.com

64% of the total websites that existed is legitimate. Out of this 64%, 74.1% advertised 'Domain Sale' . One compliment information is that we have found out that domain sale add is very prominent around popular websites. 12.3% referred to chase itself and remaining 13.6% contains other legitimate site like tech consultant, online shop and financial solutions etc.

One of our prominent observations is that we have found a lot of websites among the Typosquatting suspicious. Suspicious in the sense that they carry the logo of one website but contain a lot of URLs redirected to some other websites. This type of websites may be the hit website for click farmers carrying traffic or trying to be

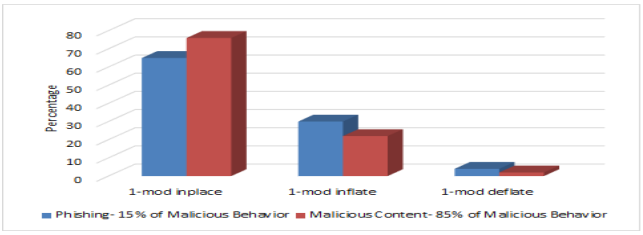


Figure 3: Forensic results: Phishing website and website with Malicious content percentage in Website showing malicious behavior in 1-mod-inflate, 1-mod-inplace and 1-mod-deflate

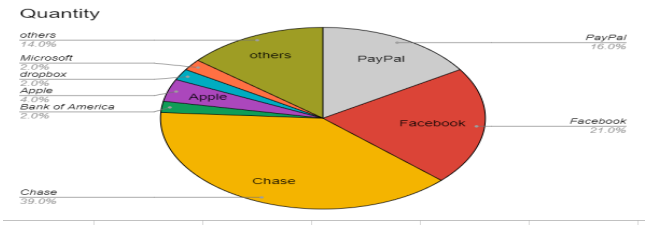


Figure 4: Forensic results: Percentage of different phishing websites in total websites analyzed from Phishtank.com

popular. Some of them are trying to sell that domain. So, we move on to our next observation 2.

Observation 2: 22.6% of the existing Typosquatts of chase.com is suspicious. That means this 22.6% website has a huge chance to be activated as malicious in near future. Out of this 22.6%, 15% is 1-mod-inflation, 2% 1-mod-deflation and the rest 83% is 1-mod-inplace. These suspicious websites have very lower page size (<15KB) and contains suspicious abnormal information. 51% of the suspicious website carry bank information like credit card information links , account opening links, homepage links of various legitimate banks as well as links from chase. May be they are trying to be trusted for now or they are just trying to increase their traffic/website hit. These websites have higher link number, invalidated HTTPS, higher URL of anchor, lower page size etc. Our inter-category features as well as combination of features from two or more categories can easily detect these suspicious websites.

Observation 3: 13.4% of the existing URL Typosquatting of chase.com website is malicious in behavior. Out of this 13.4%, 15% tries to completely imitate chase.com (actual Phishing), The rest 85% seems to have malicious content or install malicious exe. Figure 2 shows different types of Phishing tasks and their percentages. Out of the 15% phishing website, 25.2% wanted the password for login, 35.8% wanted to run survey and steal the sensitive information, 39% wanted the credit card information.

Figure 3 depicts that out of the 15% Phishing website, 65% comes from 1-mod-inplace, 30% from 1-mod-inflation and 5% from 1-mod-deflation. At the same time, it also shows that out of the 85% of

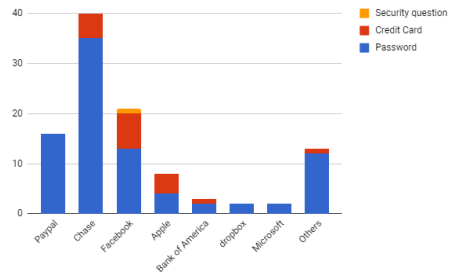


Figure 5: Breakdown of Types of actions performed by 100 phishing websites

malicious content, 76% comes from 1-mod-inplace, 22% from 1-mod-inflation and 2% from 1-mod-deflation.

4.3 Analyzing 100 Phishing Websites from PhishTank.com

At present, social and banking sites are main targets of phishing. We have analyzed 100 websites from the PhishTank.com [21] to get the idea about overall phishing target. We have visited every site and analyzed their behaviors and attributes. Among 100 websites, we have found 57 sites which are the phishing sites of some legitimate banking websites, and 21 sites are phishing sites of facebook.com. Table 3 shows the overall findings.

Table 3: Summary result of analysis of 100 reported phishing website from Phishtank.com

Website	Quantity	Password	Credit Card Info	HTTPS	Security Question
Paypal	16	16	0	5	0
Facebook	21	13	7	7	1
Chase	39	35	5	0	0
Apple	4	4	4	0	0
Dropbox	2	2	0	0	0
Microsoft	2	2	0	0	0
Bank of America	2	2	1	0	0
Others	14	12	1	2	0

Most of the phishing sites are impersonating some helping behavior of these legitimate sites so that they can fool the novice user. 20% of these websites are having very long URL and their names are very close to legitimate websites. Interestingly, 6% websites are using Google sites to impersonate other websites by customizing the URL. As Google site has trustworthy https, it is very hard to figure out that this websites are phishing. In this case, name based feature i.e. URL based features do not help that much. So, content and internet/reputation based features comes forward.

General Observation:

- 57% of phishing websites are impersonating banking websites. Among them, 39% sites are Chase bank, and 16% are PayPal. 100% phishing sites of PayPal are saving password, and 31.25% of them are using HTTPS. 89.74% phishing sites of Chase bank are saving password, and 14.29% sites are saving credit card information.
- 21% sites are impersonating Facebook. Among them, 61.9% sites are saving users passwords, 33.33% sites are saving credit card information, and 33.33% sites have valid https. 28.57% of HTTPS are using Google site.
- In total, 86% phishing sites are storing password, 18% sites are storing credit card information, 14% sites are using HTTPS, and only one phishing site of Facebook is taking security question from victims.

Figure 4 and 5 summarizes the findings. After visiting these 100 websites, we have found that most of phishing sites are having phishing attributes in their URL and content. There are 22% websites having very long URL, and most of them are typosquatting of legitimate websites. Most of the cases, URLs are very abnormal to read and having '@','-' . Even they are using HTTPS though some of them are using invalid or less trustworthy SSL certification. Domain age of these websites are very low which is an important reputation based feature in detecting phishing sites. Some sites do not let the user to right click on the webpage, and contain valid content but redirect to different domain. In other words, these websites have different kinds of phishing features. So, if we design an approach using different combination of URL based, Reputation based, and Content based instead of using all the features together, it will give better result and take less time at the same time as it will use less feature.

5 OUR APPROACH

5.1 Modules

Can we detect Phishing website with features/properties from different categories mentioned in Table 4?

From our forensics we have learned that now-a-days phishing websites are changing their appearance and properties. For example, some phishing websites can be detected only by looking at the URL names. But now they are changing their URL. So more feature categories like their reputation in the internet and their contents should be analyzed in order to detect them efficiently. For this reason, we envision RPhish as having ten modules. One module, the 'Feature Module' is to collect the features of the website that the user visits. And each of other nine modules (Module A, B, C, AB, AC, BC, ABC, CompressiveABC and X) can detect Phishing independently using the features collected from the Feature Module depending on the selection of the module by the user. Users may switch between any module. Each module uses some subset of 30 features collected by the Feature Module. For understanding the full system completely, we will first show the categorization of the 30 features in Table 4. Note that for the prediction, we have tried 9 machine learning algorithms like KNN, SVM, RF, Naive Bayes, lInear Regression, Logistic Regression, CART, BART and C48. But only the first threes exhibited mention worthy results. So, from now on, we will show the results only for these three.

Module A:

This module uses features from category A provided by the 'Feature Module' . We envision that when an user selects Module A,

Table 4: Category of features taken by "Feature Module".

Category A (Name Based)	Category B (Network /Reputation Based)	Category C (Content Based)	Inter-Category Features
Has IP Address?	HTTPS Issuer Trusted & Age of Certification	Favicon	Links outside of the domain
Long URL?	Domain Registration Length	Request URL by Image/Video	"\\" Redirection
Short URL?	Using Non-standard Port?	URL of anchor	Domain Age
Has @ Symbol?	Abnormal URL (WHOIS Search)?	# of Links in <meta>, <script> and <link>	URL of Anchor
"\\" Redirection	Age of Domain (WHOIS)	SFS Handler	HTTPS Certification
Has "-" Symbol?	DNS Record	Submitting Information to email	
Subdomain & Multi-domain status	Pagerank	Status Bar Customization	
Has HTTPS?	Website Traffic Rank	Website Forwarding	
	Google Index	Disabled Right Click?	
	Number of Links pointing to that page	IFrame Redirection	
	Statistical Report	Use Pop-Up Window?	

then Feature Module provides features of Category A and Module A builds machine learning models depending on these features and predicts the outcome. Note that features from Category A solely comprises of URL based features. A brief description of all the features from all the categories can be found in [25]. So, this is the easiest way to have the features and predict it within a short time with reasonable accuracy (90%). The problem is that when we are using only URL based features, it becomes risky as the false positive and true positive rate are relatively high. So when the user thinks that he does not want to be too judgmental about the sites whether they are phishy he can choose this Module A. Figure 6 shows the accuracy for this module

Module B:

This Module uses Network/ Reputation based features from category B provided by Feature Module. From our observation stated in section 4, we have found out that now-a-days, the phishing attacker makes phishing website with URL closer to legitimate site (Typoquatting). Module B can help in this case as the reputation of the Phishing website is not good in different sites like ALEXA.com, PhishTank.com, Malware.com and Google. The problem is that sometimes the attacker wait too long to gain the trust of the user like described above. In that case we may not be so sure about the phishing site as the accuracy is only 78% (still captures certain phishing website) but in that case we have Module C.

Module C:

As stated earlier, sometimes the attacker wait too long to acquire the trust by having some traffic. After a certain period, it triggers itself as phishing website. These kind of Phishing websites can be detected by our Module C with accuracy 90%. The problem is that it is not that usual to find this kind of Phishing website; still it helps to find the Phishing websites in certain cases. Disabling right click, Iframe redirection, showing pop-up window etc are special kinds of features used in this module.

Module AB, AC, BC:

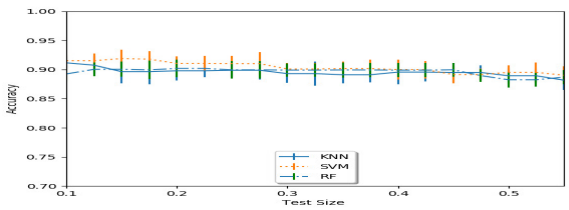


Figure 6: Test size vs accuracy for module A for KNN, SVM, RF. Performance of Module A, AB and X. Note that test size = 0.2 means 20% of the dataset was taken as test set. K=5 and number of tree in RF = 100

When user wanted to be assured about the phony behavior of a website, he can use these modules. By combining the features from two categories, users can be more confident about the Phishing detection. The increased number of features helps to detect Phishing with more accuracy. For example, Module AC uses features from Category A and Category C which actually deal with URL/name based features and Content based features. Combining these features can give us more realistic prediction with good accuracy. Figure 7 shows it's F-Score. In the Evaluation section, all the results of the modules stated above and below are going to be summarized in Table 9.

Module X:

Can we detect click farms beside of phishing?

When we want to find the Phishing sites as well as click farms, we have proposed this Module X of RPhish which uses special features from different feature Categories. From our analysis of URL Typosquatting and Phishing websites in Forensics section, we have found out these features prominent in click farms which may end up in phishing websites in near future. This module uses

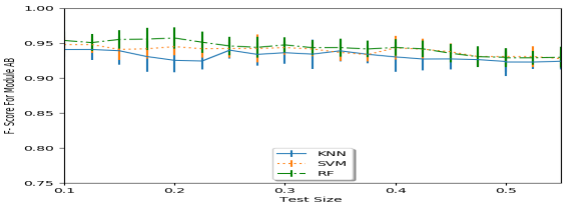


Figure 7: Test size vs F-Score for module AB for KNN, SVM, RF. Performance of Module A, AB and X. Note that test size = 0.2 means 20% of the dataset was taken as test set. K=5 and number of tree in RF = 100

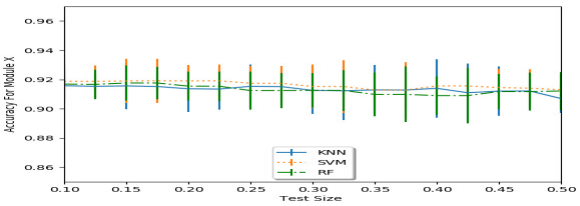


Figure 8: Test size vs Accuracy for module X for KNN, SVM, RF. Performance of Module A, AB and X. Note that test size = 0.2 means 20% of the dataset was taken as test set. K=5 and number of tree in RF = 100

inter-category features supplied by the Feature Module. Figure 8 demonstrates the performance of this module.

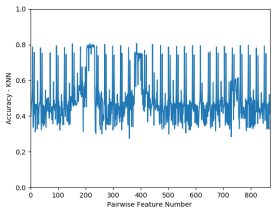
Module ABC:

This module uses all the features from all three categories. So in total, there are 30 features. Using these higher number of features along with our larger dataset can be very accurate in predicting Phishing. The problem with large feature size is its slow prediction time. But if the user wanted to be more ensured about the outcome for a sensitive task, he can surely use this module. In fact we will recommend to use large feature size as for having good accuracy, high true positive rate, and low false positive rate.

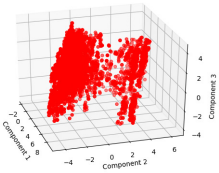
5.2 Feature Compression

Can we detect phishing website with faster prediction time and higher accuracy reliably?

The answer to this above question is- yes. If we want to detect phishing websites with confidence, we will have to analyze more features instead of only limited number of features. But if we take all the 30 features in our hand in account, we will get high accuracy but we could end up higher prediction time. So we have taken all 30 features but successively compressed/mapped them in lower dimension. This has drastically improved the prediction time and also the accuracy. Note that the processor time shares all the jobs in its hand and also the internet connection of the user may usually very slow. So, improving the prediction time even by a factor of two can be very useful as users want to spend minimum amount of time in extra works besides loading the web pages. In this situation, our compressive modules can come in a big help. Again the features



(a) Accuracy of KNN for distinct pairwise features



(b) PCA Components after PCA compression. # of component = 3. Note that after compressing in three components, the data are easily distinguishable by a single plane

Figure 9: Intuition behind feature compression

that we are working with are given in section Table 4. We have taken all features from Category A,B,C.

Accuracy of Triplet/ Pairs:

We have analyzed our features taking them in pairs and also in triplets. We have found out that none of the pairs or the triplets crossed 82% accuracy for KNN. But we have got an important notion that some features are very important- thus PCA or other feature compression algorithms can show interesting results.

PCA:

PCA actually compresses the features and project them in some other compressed dimension where the data points shows maximum variance. The instances may not be separable in original dimensions but may be separable in some other dimension even if the features are compressed if the data points shows some specific properties. Figure 9 shows the intuition for feature compression.

Gaussian Random Projection and Sparse Random Projection:

Random Projection is a simple and computationally efficient way to reduce the dimensionality of the data by trading a controlled amount of accuracy (as additional variance) for faster processing times and smaller model sizes. Random Projection is a suitable approximation technique for distance based method [24]. The main theoretical result behind the efficiency of random projection is the Johnson-Lindenstrauss lemma concerning low-distortion mappings of points from high-dimensional into low-dimensional Euclidean Space. We have applied two types of Random Projection- Gaussian Random Projection and Sparse Random Projection on our dataset. Note that this Random Projection methods are being newly applied on this dataset and also on phishing detection.

Module CompressiveABC:

The problem with Module ABC is that it takes long prediction time though it shows highest accuracy. So if the user is in a hurry but still he wants good accuracy, then we have a solution for that. We have applied compression based algorithms like PCA, Gaussian Random Projection and Sparse Random Projection on this 30 features and then modeled it in data mining algorithms like KNN, SVM, Random forest etc. We have been able to detect a phishing website using this compressive model even better than the non-compressive version with greater accuracy and surely with lowest prediction time.

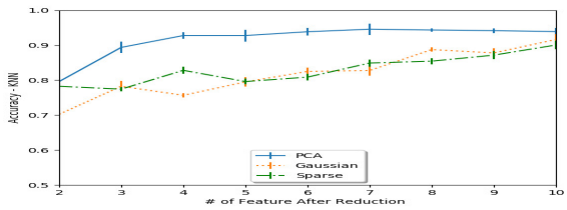


Figure 10: Compressive ABC a) Accuracy of KNN vs Number of features after compression by PCA,Gaussian and Sparse Random Projection on 30 features

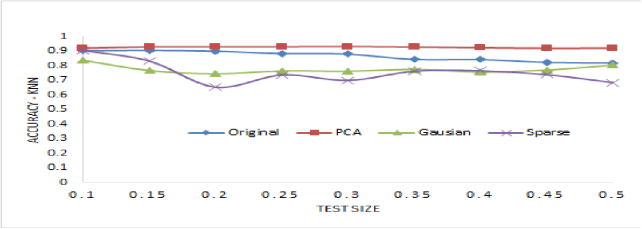


Figure 11: Compressive ABC b) Test size vs Accuracy of KNN for Original Module (Module ABC), compression by PCA, Gaussian and Sparse Random Projection

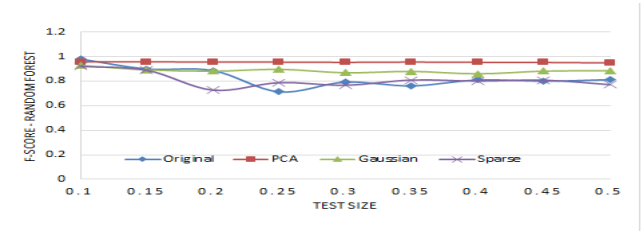


Figure 12: Compressive ABC c) Test size vs F-Score of RF for Original Module (Module ABC), compression by PCA, Gaussian and Sparse Random Projection

We have found out that PCA works the best among the three dimension reduction algorithms. It gives us better accuracy, TPR, ROC-AUC, F-Score and prediction time than Module ABC. Figure ??, 11, 12 demonstrates these results.

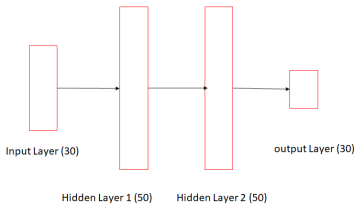


Figure 13: Architecture of the Module 'Deep'

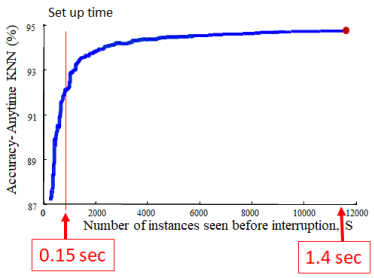


Figure 14: Performance of Anytime Version of KNN. (k=5)

5.3 Neural Network

Can we acquire more and more accuracy with less prediction time?

The result to the above question is- yes, we can acquire more and more accuracy if we apply certain architecture of the Deep Neural Network. We call this module-Module Deep.

Module Deep:

If the users want to focus on accuracy about the detection, he can apply this module. However, there are some issues here.1) The higher accuracy may cause overfitting. 2) Normally DNN works very fine for huge data and here we have only 11000 instances. We will address these issues here. For of these issues to be resolved, we have used validation set accuracy as a measure to be sure that the dataset is not suffering from overfitting or less data. We have found an average of 96% validation set accuracy and maximum 98% test set accuracy in almost minimum epoch 10. The architecture is given in Figure 13 and the performance is shown in Figure 17 : As the architecture is not that deep (like multilayer perceptron), we have gained a very good prediction time of 0.00000875 second for a single instance. This in very lower than the traditional machine learning algorithms. Not only this DNN architecture but also we have tried various neural network architectures but we could not get that good result like this DNN architecture.We will summarize our results and parameters in Evaluation section.

5.4 Majority vote and Expert/User choice

In this phase, we have built a new module which is the fusion of some former module. This new fusion module will be called Module Majority Vote: Also we have built some other modules which will work based on usersćhoice as we envision this RPhish tool to be interacting with the users as a browser integration or software-thus engaging the users in a interactive and fruitful manner in phishing detection.

Module Majority Vote:

In this module, we have gathered all the predicted results from Module A,AB, ABC, Compressed ABC and Module Deep . Then we have taken the majority votes of these modules. This majority vote module shows good accuracy (96.11% usingr SVM in all modules) since it combines the results of five distinct modules. As a result, we can have much more reliable prediction. The problem is that the user has to wait a bit for the confirmation whether the web-site is Phishing. The table below exhibits a glimpse of the Module (L=Legitimate and P=Phishing).

Module myFeature:

Table 5: Category of features taken by "Feature Module"

Test Set Touple	Module A	Module AB	Module ABC	Compressed ABC	Module Deep	Majority Vote	Actual Label
1	L	L	L	L	L	L	L
2	P	L	L	L	L	L	L
3	P	P	L	P	P	P	P
4	L	P	P	P	P	P	P
5	L	P	L	L	P	L (Wrong)	P

This module is for the specialist users to have a knowledge of the dataset, website and RPhish. If the expert users think that some features/algorithms may be very important to detect phishing, he/she can easily set his own feature set/algorithm used in this paper.

Module Anytime:

Can we predict the outcome whether the website is phishing ahead of the algorithm's completion time?

Yes. we can predict the outcome ahead of the completion of the full algorithmic run time using anytime algorithm. This gives the user the advantage of interrupting the computation and giving at least some results ahead of the finishing. Sometimes users are in a hurry and want to have at least a reasonable result. In this situation we can use anytime algorithm. But the thing is- not every algorithm can be converted to anytime algorithm. Among our choices, we have converted KNN to Anytime algorithm by comparing the instances in our dataset with the one whose label we are going to predict before the user interrupts. To remove the problem of comparing with same types of instances we have first reorganized the dataset with alternative instances of phishing and legitimate labeling. As there is no redundant instances in the dataset, using this primary setting up/rearrangement of the dataset, we can have a very good result. The primary set up time includes comparison with a minimum number of instances (500) to have at least a result after the set up time. In this set up time, the users can not interrupt. Figure 14 shows that this anytime module can give us reasonable result and actually give better result if given more and more time. This Anytime algorithm is novel in this domain.

5.5 Feature Selection

Can we find the minimum set of the most important and influential features to detect phishing website?

The answer to the above question is - yes. There are some features in this dataset that can be very influential who have maximum impact on the classification. One thing to mention that these minimum feature set solely can give good accuracy(90%)but can suffer from bad performance for following unseen data. But we may get a reasonable accuracy with fast prediction time. However, to find out the most important features, we have run three algorithms on our dataset- Forward Selection, Backward Elimination and Correlation based Filter method. Actually we have modified the first two algorithms to find out the minimal set of important features with a certain accuracy as well as their final outcome which gives us the feature set with highest accuracy. Note that we have used SVM algorithm to find out the accuracy and the threshold was 90%-92% with a step size of 1%. Using Correlation based Filter method, we have found the most influential features which are highly correlated with the class label. We have tested with Pearson and Spearman Correlation Coefficient method to find the highly correlated features with the class label- both giving the same results for the first 11 features and then used Modified Forward Selection. Algorithm 1 gives us a clear insight of this method. The summarized results are shown in Table 6, 7 and Figure 15 :

Table 6: Important feature extraction using forward selection algorithm

Feature Selection Algorithm	Feature Set (Accuracy-90%)	Feature Set (Accuracy-91%)	Feature Set (Accuracy-92%)	Minimum # of features for max accuracy
Forward Selection	SSL Final State, URL of Anchor	SSL Final State, URL of Anchor, Link in Tags	SSL Final State, URL of Anchor, Link in Tags, Redirection	14

Result: Print the important feature set with "threshold" and maximum accuracy
sort the features in decreasing order of Pearson correlation value between features and the label in sorted_list;
current_set = {};
while *level in range 1 to total_feature* **do**
  f = Remove a feature from the front of the list with highest correlation value;
  current_set = current_set U f;
  accuracy = measure the accuracy with "current_set";
  map = store accuracy and current_set as key and value pair;
  if *accuracy greater than "threshold" && condition satisfied for the first time* **then**
    Print current_set and accuracy;
  end
end
find the best accuracy and feature set from map;

Algorithm 1: Modified correlation based filter method for feature selection

5.6 Dataset

We collect our dataset from UCI Dataset Archive [12] which contains 30 features like URL length, number of redirection, requested URL, URL of anchor etc. and labeled the touples as phishing or legitimate for 11055 instances. The dataset is much more balanced as itś default accuracy is 50.56%. The dataset was donated on July 11, 2015. So it is comparatively new data.

In this section, we will evaluate our modules, compare them with one another and existing others. We will focus on prediction time

Table 7: Table to show the important features found by the modified correlation based filter method

Algorithm name	Top 5 correlation features	correlation coefficient value (Pearson)	Max accuracy with these 5 features
Modified Correlation based Filter method	SSL Final State	0.714	0.870
	URL of Anchor	0.692	
	web traffic	0.340	
	Link in Tags	0.253	
	Request URL	0.240	

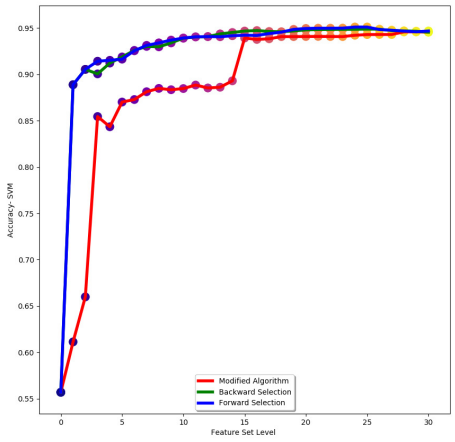


Figure 15: Feature Selection algorithms's outcome. The figure describes that at each label, x, the current feature set will contain x number of features with accuracy y.

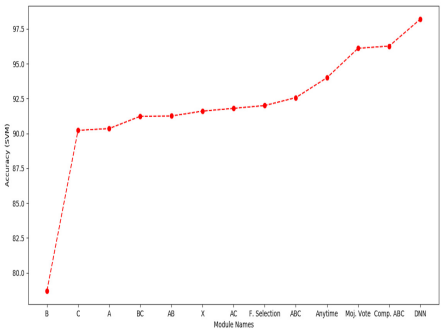


Figure 16: Summary of the accuracy of the Modules used in RPhish. Note that the Feature Selection module accuracy is for 4 features and Anytime module accuracy is for 4000 instances.

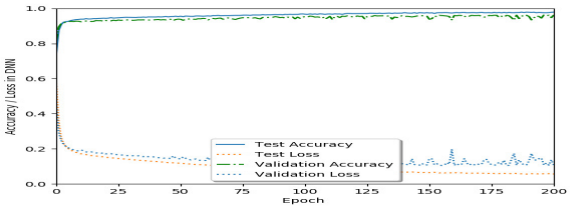


Figure 17: a) Performance matrices of the Module 'Deep'

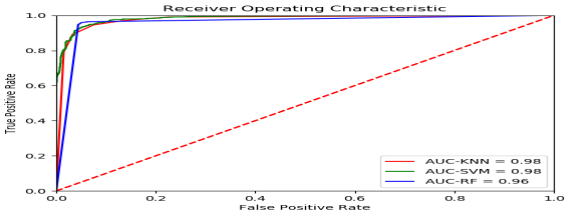


Figure 18: b) ROC AUC of dataset with original features (Module ABC)

which got no attention in previous work. Also we will evaluate our models and approaches using general matrices that existing works have used. We will also talk about the time complexity for feature compression in this section. Note that we have measured almost all the matrices using 10-fold cross validation. So the results shown in this section is very stable. We have performed all the computations in intel(R) CORE i5(TM)-5200U, 2.2 GHz processor with 8GB RAM, 64 bit Operating System.

Prediction Time

Prediction time is one of our unique point of views depending on what we have analyzed our modules. Our target is acquiring best accuracy with least prediction time. In figure 17, we have analyzed this result. For getting best accuracy , we must use all 30 features from the dataset and for getting least prediction time, PCA is best fit for it. We have got least prediction time for KNN, RF and SVM. Using PCA with these algorithms we have got less prediction time as well as best accuracy than without any compression. Figure 19 demonstrates that if we use original feature set we will end up getting high prediction time. For example, using KNN we have got 1.4 second single prediction time. But if we compress the features in 3 or 4 component, the single prediction time becomes 0.2 second which is an improvement with a factor of 7. Note that one may think that this 1.4 second prediction time is not that much time. Moreover, using RF with original 30 features, we may get 0.65 second prediction time. But if we think about the general user's satisfaction, CPU's other tasks and low computation power and bad internet connection, we will understand its usefulness. Our approach can lessen the prediction time by a factor of 7 for KNN if we compress the 30 dimension to 4 dimension and it also surpasses the current existing best accuracy. This is a huge improvement. The prediction time for SVM with PCA is not that good as KNN + PCA. But still it is an improvement. Now the next question is:

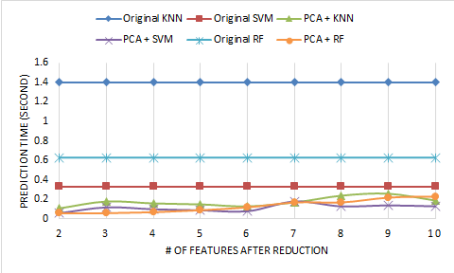


Figure 19: Number of components after compression vs prediction time for Module ABC ('Original' in graph)+KNN, ABC with SVM, PCA on 30 features+KNN , PCA+SVM

'Can we get better prediction time than this above approach?'. The answer to this question is - yes. Using particular architecture of Deep Neural Network (DNN), we can acquire least prediction time once we train our data and save the weights and biases. The improvement is huge because we can improve the prediction time by a factor of 160000 as the architecture is not that deep (considering prediction time of Module ABC with KNN= 1.4 seconds). We have also tried other neural network architectures on our dataset. But the DNN works the best yielding best accuracy and best prediction time. Table 8 summarizes the result. Note that we have tried many variations/ architectures of neural network with different parameter combinations. Only we have stated of those who showed the best results.

Accuracy, TPR, FPR and F-Score

Accuracy, TPR, FPR, F-Score are the general matrices that previous works used for phishing detection classification problem. So, our goal of this work was to detect phishing websites with best accuracy, TPR, FPR and F-Score. We have measured these matrices for all the modules on our dataset, compared their results with the existing ones and found out the best result in almost all aspects. Table 9, 8, 10 and Figure 16 summarize our findings. Note that he Default Rate for this binary classification problem on our dataset is 50.56%. This proves that our data is very much balanced.

ROC AUC:

Figure 18 and **??** demonstrate the ROC curve for Module ABC and Module Compressive ABC respectively. These figures prove that our compressive module have got a very good TPR and FPR and these matrices is not affected by the compression. And the AUC value for PCA + RF is: 0.99 compared to 0.96 without any compression. This is also an indicator of good result for us. The AUC value of Module Depp is 0.99.

Time Complexity of prediction:

In this part, we will try to analyze the time complexity of our Module Compressive ABC.

Let, total instance= S

We define compression ratio, R= # of original features (N)/# of features after compression (M)

- PCA + KNN: O(SN/R)
- PCA + SVM: O(# of support vector in (N/R) * (N/R))
- PCA + RF: O(# of tree * N/R * S *log S)

The example/result of this compression ratio is stated in Prediction time paragraph.

Now, among these three algorithms- KNN, SVM, RF, no algorithm is good in every aspect. So, we are not suggesting any sole algorithm for every module. Rather we envision our RPhish as having options for all three algorithms with recommendation of using certatin algorithm for certain module and letting the users fix their choice.

6 FUTURE WORKS:

Our work has focused on improving prediction time with good accuracy. One may argue that the feature size is not that high, so the gain may not that significant. But here we want to mention that our dataset size and also the feature size is relative high than any others dataset that had been worked on. Also for the lazy algorithms like KNN and Random forest, our approach suits well. The models of the Machine Learning techniques for our big dataset becomes compressed for feature compression and gives a significant improvement for real time prediction specially the KNN. We still want to explore the effect of our technique for greater feature size. That is why our feature work includes applying the same compressive technique on P.A. Barraclough et al.'s [6] works for Phishing detection with more features (287). We also have a plan to apply our approaches on phishing email detection. However, in our works, we are still looking forward to finding the frequent feature set that comes in phishing detection.

7 CONCLUSION

We have introduced feature compression methods like PCA, Gaussian Random Projection, Sparse Randomr Projection on our phishing dataset and then applied 9 machine learning classification algorithms. Then we have applied different architectures from Neural Networks like DNN, CNN, LSTM, RNN, Autoencoders etc and for the users we have introduced Anytime Algorithm. Also we have found the important feature set in our data set. These approaches and techniques have made RPhish more robust, scalable, accurate with best prediction time and can come in great help for the internet community saving them from the hazard of phishing alltack.

REFERENCES

[1] Saeed Abu-Nimeh, Dario Nappa, Xinlei Wang, and Suku Nair. 2007. A comparison of machine learning techniques for phishing detection. In *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*. ACM, 60–69.

[2] Maher Aburrous, M Alamgir Hossain, Keshav Dahal, and Fadi Thabtah. 2010. Intelligent phishing detection system for e-banking using fuzzy data mining. *Expert systems with applications* 37, 12 (2010), 7913–7921.

[3] Maher Aburrous, M Alamgir Hossain, Keshav Dahal, and Fadi Thabtah. 2010. Predicting phishing websites using classification mining techniques with experimental case studies. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*. IEEE, 176–181.

[4] Anti-Phishing Working Group (APWG). 2017. (Dec. 2017). Retrieved Dec 20, 2017 from http://www.antiphishing.org/

[5] Anirban Banerjee, Md Sazzadur Rahman, and Michalis Faloutsos. 2011. SUT: Quantifying and mitigating url typosquatting. *Computer Networks* 55, 13 (2011), 3001–3014.

[6] PA Barraclough, M Alamgir Hossain, MA Tahir, Graham Sexton, and Nauman Aslam. 2013. Intelligent phishing detection and protection scheme for online transactions. *Expert Systems with Applications* 40, 11 (2013), 4697–4706.

[7] Ram Basnet, Srinivas Mukkamala, and Andrew H Sung. 2008. Detection of phishing attacks: A machine learning approach. In *Soft Computing Applications in Industry*. Springer, 373–383.

[8] Federal Trade Commission. 2017. An E-Card for You game. (Dec. 2017). Retrieved Dec 20, 2017 from http://www.ftc.gov/bcp/conline/ecards/phishing/index.html

Table 8: Result evaluation of our dataset with Neural Network. Epochs=200 and batch size= 128/256,categorical cross entropy, relu for DNN dense layers, one-hot, 10-fold cross validation, test size=0.2, validation size=0.2, training size= 0.6, prediction time function= timeit.default_timer() of python3, 10-fold cross validation and number of features= 30.

Architecture/ Module Name	Parameters	Layers	Test Set Accuracy(%)	Prediction time (second)
Module ABC (SVM)	Cache size- 200, probability- false, shrinking- True, kernel- rbf	N/A	92.57	0.399
Module Compressive ABC (SVM)	PCA: n_component- 4, copy -True, svd solver- auto. SVM: Cache size- 200, probability- false, shrinking- True, kernel- rbf	N/A	96.27	0.071
Module Deep (DNN + Module ABC)	sigmoid, sgd	30-50-50-2	98.20	0.00000875
Module Deep	sigmoid	30-50-50-50-2	98.20	0.00009875
Module Deep	sigmoid	30-100-100-2	96.99	0.0009
Module Deep	sigmoid	30-50-2	93.20	0.00000575
DNN + PCA	DNN: sigmoid, sgd PCA: n_component-5	30-50-50-2	90.45	0.00000575
DNN + PCA	DNN: sigmoid, sgd. PCA: n_component- 10	30-50-50-2	91.28	0.00000746
DNN + PCA	sigmoid, sgd, n_component-15	30-50-50-2	92.14	0.0000079
DNN + Gaussian Random Projection	sigmoid, sgd, n_component- 10	30-50-50-2	89.47	0.00001
CNN	softmax, maxpool, adam, poolsize- 2*2	Convolution layer: 64,128,128,128, 256,256,256 Fully connected: 1024,2	90.43	0.000108
LSTM	Drop out rate- 0.2, rmsprop, output activation- softmax	30-100-100-2	90.14	0.000403
RNN	rmsprop, output activation- softmax		91.96	0.0004
Deep Autoencoder+ DNN	Autoencode: output activation- sigmoid, binary cross entropy, adadelta DNN: sigmoid	Autoencoder layers: 30-24-16-10-16-24-30 DNN layer: 30-50-50-2	94.66	0.0009

Table 9: Table: Accuracy comparison among different Data mining algorithms and existing best accuracy

Module Name	Accuracy KNN	Accuracy SVM	Accuracy RF	Existing Best
A	89.537	90.347	90.01	95
B	77.11	78.71	79.09	
C	89.753	90.225	91.235	
AB	91.318	91.25	91.7	
AC	91.401	91.8	92.111	
BC	90.5	91.234	91.667	
ABC	89.3	92.57	92.733	
Compressive ABC	94.188	96.27	95.313	
X	91.5	91.6	91.68	
Majority Vote	95.778	96.11	94.125	

Table 10: Comparision between existing best accuracy, TPR,FPR, F-Score and Module 'Compressive ABC' . Note that current best accuracy, TPR and FPR (without human intervention) is in[27]

Name	Our Best Accuracy	Current Best Accuracy	Our Best TPR	Current Best TPR	Our Best FPR	Current Best FPR	Our Best F-Score	Current Best F-Score
Compressive ABC	96.27	95	97.6	97	4.3	3	98	91
Module Deep	98.21		98.64		2.3		98.13	

[9] Federal Trade Commission. 2017. Phishing Alerts. (Dec. 2017). Retrieved Dec 20, 2017 from http://www.ftc.gov/bcp/conline/pubs/alerts/phishingalrt.htm

[10] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20, 3 (1995), 273–297.

[11] Rachna Dhamija, J Doug Tygar, and Marti Hearst. 2006. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM, 581–590.

[12] Dua Dheeru and Efi Karra Taniskidou. 2017. UCI Machine Learning Repository. (2017). http://archive.ics.uci.edu/ml

[13] eBay Inc. 2017. Spoof Email Tutorial. (Dec. 2017). Retrieved Dec 20, 2017 from http://pages.ebay.com/education/spooftutorial/

[14] Gregg Keizer. 2005. Phishing costs nearly $1 billion. *TechWeb News* 6, 24 (2005), 05.

[15] Rober McMillan. 2006. Gartner: Consumers to lose $2.8 billion to phishers in 2006. *Computer World* (2006).

[16] Microsoft. 2017. Consumer Awareness Page on Phishing. (Dec. 2017). Retrieved Dec 20, 2017 from http://www.microsoft.com/athome/security/email/phishing.mspx

[17] Daisuke Miyamoto, Hiroaki Hazeyama, and Youki Kadobayashi. 2008. An evaluation of machine learning-based methods for detection of phishing sites. In *International Conference on Neural Information Processing*. Springer, 539–546.

[18] Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. 2012. An assessment of features related to phishing websites using an automated technique. In *Internet Technology And Secured Transactions, 2012 International Conference for*. IEEE, 492–497.

[19] Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. 2014. Predicting phishing websites based on self-structuring neural network. *Neural Computing and Applications* 25, 2 (2014), 443–458.

[20] Ying Pan and Xuhua Ding. 2006. Anomaly based web phishing page detection. In *Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual*. IEEE, 381–392.

[21] PhishTank. 2017. Phishing sites published in PhishTank.com. (2017). Retrieved December 30, 2017 from http://data.phishtank.com/data/online-valid.csv

[22] Issa Qabajeh and Fadi Thabtah. 2014. An experimental study for assessing email classification attributes using feature selection methods. In *Advanced Computer Science Applications and Technologies (ACSAT), 2014 3rd International Conference on*. IEEE, 125–132.

[23] Nuttapong Sanglerdsinlapachai and Arnon Rungsawang. 2010. Using domain top-page similarity feature in machine learning-based web phishing detection. In *Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on*. IEEE, 187–190.

[24] Scikit-learn.org. 2018. Random Projection. (2018). Retrieved March 20, 2018 from http://scikit-learn.org/stable/modules/random_projection.html

[25] UCI. 2018. Phishing features set and feature description. (2018). Retrieved March 10, 2018 from https://archive.ics.uci.edu/ml/machine-learning-databases/00327/

[26] Guang Xiang, Jason Hong, Carolyn P Rose, and Lorrie Cranor. 2011. Cantina+: A feature-rich machine learning framework for detecting phishing web sites. *ACM Transactions on Information and System Security (TISSEC)* 14, 2 (2011), 21.

[27] J Hong YZhang and L Cranor. CANTINA: A content based approach to detect phishing websites. In *Proceedings of the 16th International Conference on World Wide Web, Banff, AB, Canada*. 639–64S.