Center for Open Access in Science ▪ https://www.centerprode.com/ojit.html
Open Journal for Information Technology, 2023, *6*(2), 123-136.
ISSN (Online) 2620-0627 ▪ https://doi.org/10.32591/coas.ojit.0602.03123n

Detecting Phishing Emails Using Random Forest and AdaBoost Classifier Model

Fredrick Nthurima, Abraham Mutua & Waithaka Stephen Titus

Kenyatta University, School of Pure and Applied Science, Nairobi, KENYA

Received: 7 July 2023 ▪ Revised: 4 October 2023 ▪ Accepted: 15 November 2023

Abstract

Phishing attack occurs when a phishing email which is a legitimate-looking email, designed to lure the recipient into believing that it is a genuine email to open and click malicious links embedded into the email. This leads to user reveal sensitive information such as credit card number, usernames or passwords to the attacker thereby gaining entry into the compromised account. Online surveys have put phishing attack as the leading attack for web content mostly targeting financial institutions. According to a survey conducted by Ponemon Institute LLC 2017, the loss due to phishing attack is about $1.5 billion per year. This is a global threat to information security and it's on the rise due to IoT (Internet of Things) and thus requires a better phishing detection mechanism to mitigate these loses and reputation injury. This research paper explores and reports the use of a combination of machine learning algorithms; Random Forest and AdaBoost and use of more phishing email features in improving the accuracy of phishing detection and prevention. This project will explore the existing phishing methods, investigate the effect of combining two machine learning algorithms to detect and prevent phishing attacks, design and develop a supervised classifier which can detect phishing and prevent phishing emails and test the model with existing data. A dataset consisting of both benign and phishing emails will be used to conduct a supervised learning by the model. Expected accuracy is 99.9%, False Negative (FN) and False Positive (FP) rates of 0.1% and below.

Keywords: classification, algorithm, cyber security, machine learning, spam emails, cyber security, cyberattack, web attacks, intrusion detection and phishing emails, AdaBoost, Random Forest.

1. Introduction

1.1 *Background to the study*

According to Anti-Phishing Working Group report 2018, Phishing attack is the number one attack committed by threat actors as compared to other attacks. It is a form of fraud where the attacker deceives the target for personal gain or reputation damage. Fraud results to users revealing their personal details like credit card numbers, passwords, PIN, usernames and other sensitive information leading to compromise of account and loss of funds.

Phishing campaign lures users to giving confidential information by visiting websites that have been made to look like legitimate websites (phishing.org, 2018). Phishing is carried out using a digital garget like a computer or iPad through a computer network. Malicious actors usually target weakest element in the security chain, i.e., end-users (Khonji, Jones & Iraqi, 2013).

© **Authors**. Terms and conditions of Creative Commons Attribution 4.0 International (CC BY 4.0) apply.
Correspondence: Fredrick Nthurima, Kenyatta University, School of Pure and Applied Science, Nairobi, KENYA.

Attackers in the phishing campaign usually craft messages known as social engineered messages persuading users to click and visit the illegitimate websites thus revealing their confidential information to attackers. This enables threat actors gain entry into the compromised account and achieve their objectives like data theft, funds transfer or reputation injury.

For instance, a malicious email might have a malware which when clicked by the user will install itself in the pc or mobile phone and will transfer funds to the account of the attacker whenever the owner of the account tries to transfer cash (Khonji et al., 2013). This attack is called Man in the browser (MITB) which is a variant of the Man in the middle (MITM) attack. The man in the browser attack usually uses different vectors like ActiveX components, plugins or email attachments to deliver the payload to the user's computer or phone.

With the increasing case of cyber-attacks, organizations are looking for safer ways of protecting data and prevent getting hacking or getting hacked again. Design and technology should be greatly improved to ensure hackers do not infiltrate into networks.

According to (Behdad, French, Bennamoun & Barone, 2012), using better defense systems is not enough in stopping malicious actors from penetrating systems since these are sometimes circumvented; a better system should detect malicious activities and prevent them before causing any damage.

There are a number of mechanisms used today to filter spams but they are static in nature such that they cannot handle the ever-evolving threats and phishing trends. They are only capable of detecting already known phishing patterns leaving behind future attacks. This is a security weakness because attackers are not static in nature and use different ways of evading detection. This challenge has motivated researchers into looking for other ways of detecting both known and new threats which led to the knowledge and use of machine learning algorithms.

Machine learning (ML) is a discipline of artificial intelligence that uses data mining to detect new and existing phishing features from a given dataset which is ultimately used for classification of benign and phishing emails.

In this project, we will use a combination of two ML algorithms namely random forest and AdaBoost and a set of 15 important phishing features as identified from the literature. The dataset will consist of 3000 emails from both phishing and benign sources and then extract the features for each email, form a vector representation of these extracted features which will be used to train our classifier model.

1.2 *Problem statement*

Very few phishing email filters have been developed as opposed to many existing email filters that have been developed for spam emails. Many of them used several phishing detection techniques ranging from blacklists, visual similarity, heuristic, and machine learning. Of all these techniques, ML-based technique does offer the best results (Brown, Ofoghi, Ma & Watters, 2017).

However, current machine learning anti-phishing solutions use a single algorithm to detect phishing. This according to results doesn't offer best accuracy of detection which currently stands at 98% (Smadi, Aslam, Zhang, Alasem & Hossain, 2015). Moreover, they have used domain/url characteristics leaving behind other phishing features that are present in phishing emails therefore lowering accuracy and detection rates.

There is need to investigate the use of combination of two machine learning algorithms namely Random Forest and AdaBoost and include other phishing email features to increase detection accuracy to 99.9%, fewer FPs and FNs and increase overall phishing detection and prevention.

1.3 *Objectives*

This project is aimed at achieving the following objectives:

1. To investigate the existing phishing attacks methods used by attackers to lure users.

2. To investigate the effect of combining Random Forest and AdaBoost algorithms and use of more features in phishing email detection.

3. To design and develop a supervised classifier model which can detect phishing emails.

4. To test the classifier model with existing data.

1.4 *Research questions*

1. How do attackers lure users to visit phishing websites?

2. Can the use of combination of ML algorithms and use of more features lead to increased accuracy in phishing detection?

3. To what accuracy can ML achieve phishing detection?

4. What recommendations can be inferred for future classifiers?

1.5 *Research scope*

1. This project will address phishing emails.

2. The project will use a combination of Random Forest and AdaBoost machine learning algorithms.

3. A total of 15 learning features will be used.

4. Algorithms will be implemented using python frameworks.

2. Literature review

2.1 *Introduction*

Phishing attack occurs when a phishing email which is a legitimate-looking email which is designed to lure the recipient into believing that it is a genuine email to open, and click malicious links embedded into the email. This leads to user revealing sensitive information such as credit card number, usernames or passwords to the attacker thereby gaining entry into the compromised account (Holbrook, Kumaraguru, Downs, Cranor & Sheng, 2010).

Approximately 57% of phishing attacks target financial institutions and payment services, according to Phishing Activity Trends Report – 4th Quarter 2017, Anti-Phishing Working Group (APWG).

Phishing is a widely spread threat in the Internet and is achievecd when an attacker lures a user into entering sensitive information like passwords or credit card numbers into illegitimate website that is controlled by malicious actor. It has been demonstrated that social phishing, where the word “social” means information related to the victim is used, produces very effective results compared to regular phishing. Gupta, Prakash, Kompella and Kumar (2015) found that if phishing e-mails impersonated a target’s friend, the success rate of the phishing attack

increased from 16% to 72%. The social aspect of information is therefore not only of value to social network operator but also to attackers. This is made even more possible if the information on social media contains a valid email address or there is a recent conversation between the victim and the impersonated friend.

With automation of data extraction from social media networks, a lot of usable data is available to attackers which can be used to carry out phishing attacks. Information extracted from social media networks is misused by the context-aware spam to increase appearance of authenticity of traditional spam messages.

Brown, Ofoghi, Ma and Watters (2017) classified three context-aware spam attacks: relationship-based attacks, unshared-attribute attacks, and shared-attribute attacks. Relationship-based attacks exploit relationship information only thus making this the spam equivalent of social phishing. The other two attacks exploit additional information from social networks, information that is either shared or not shared between the spam target and the spoofed friend. An example of an unshared attack are birthday cards that seem to originate from the target's friend. Shared attributes, e.g., photos in which both the spam target and the impersonated friend are tagged, can be exploited for context-aware spam. Huber, Mulazzani, Leithner, Schrittwieser, Wondracek and Weippl (2011) found that the missing support for communication security can be exploited to automatically extract personal information from online social networks. Furthermore, the authors showed that the extracted information could be misused to target a large number of users with context-aware spam.

Gupta, Prakash, Kompella and Kumar (2015) used a hybrid of two techniques namely blacklists and heuristics to detect phishing emails which achieved a FP and FN rates of 5% and 3% respectively.

Holbrook et al. (2010) conducted an investigation on some anti-phishing toolbar and reported SpoofGuard which was developed by Ledesma, Chou, Mitchell and Teraguchi (2014) to have a FP rate of 38% and a FN rate of 9%. Also, Nargundkar, Tiruthani and Yu (2017) developed a heuristics-based phishing detection system which achieved a FP and FN rates of 1% and 20%, respectively. Smadi et al. (2015) also used heuristics technique and their method achieved a FP rate of 3% and FN rate of 11%.

Sadeh, Fette and Tomasic (2017) used Machine Learning based technique and they achieved a FP rate of 1% and a FN rate of 1.2%. Strobel, Glahn, Moens, De Beer and Bergholz (2010) combined the use of heuristics and ML-technique and their method achieved a FP rate of 0.05% and a FN rate of 1%.

All these proposed methods have relatively high FP rate and FN rate except for Sadeh et al. (2017) and Strobel et al. (2010) whose techniques achieved excellent results with very low FP and FN rates. However, Strobel et al. (2010) used model-based features involving the processing of images which results in increased runtime and space. Sadeh et al. (2017) also made use of a domain name feature that has to be obtained by sending of queries over the network which results to increased run-time. In our proposed method, the phishing email features are extracted directly from the email. Thus, by eliminating sending of queries, the proposed model will be faster and remove space complexities.

2.2 *Common ml anti-phishing techniques*

Phishing emails can be classified using complex techniques based on specific features such as URL length, sub_domain, prefix_suffix and many more. Mohammad, Thabtah and McCluskey (2013) created unique learning bases making use of space understanding to detect phishing and legitimate emails. Recently, there has been many studies for achieving automated

rules to separate genuine and phishing emails with the use of statistical analysis (Abdelhamid, Thabtah & Ayesh, 2014). For instance, Mohammad, Thabtah and McCluskey (2014) grouped many intelligently derived rules in regards to different phishing features by using frequency counting of phishing emails (instances) gathered from various sources including PhishTank and Yahoo directory. Improvements in rules for decision making have been developed whereby a computational intelligence method on a larger phishing dataset collected from various sources have been used (Abdelhamid, Thabtah & Ayesh, 2014).

Phishing was studied using C4.5, decision tree, random forest, support vector machine and Naïve Bayes approaches. "Phishing Identification by Learning on Features of Email Received" (PILFER) was developed as an anti-phishing technique and then investigated on a set of 860 phishing case and 695 ham cases. The results were different features for recognising instances as phishing or ham, i.e., IP URLs, time of space, HTML messages, number of associations inside the email, JavaScript and others. Therefore, the authors explained that PILFER can improve the clustering of messages by joining all ten features found in the classifier beside "Spam filter output".

In order to reduce both false positives and false negatives an evaluation of Random Forest algorithm was conducted against 2000 messages (Akinyelu & Adewumi, 2014). After experimentations with a 15-feature dataset, the results show a reduction in error rate when using Random Forest and therefore use of this algorithm as a method for phishing classification seemed fitting. The models using Random Forest seemed to be more dominant with respect to detection rate.

Aburrous, Hossain, Dahal and Thabtah (2010) conducted another project to accurately classify websites based on features. The authors manually classified features into six criteria and then load them into an environment for analysis on Waikato Environment for Knowledge Analysis (WEKA). During this exercise, various experiments were ran using four classification algorithms against 1006 instances from PhishTank. The evaluation criteria to determine the applicability of the features was the classification accuracy. The results showed that decision tree algorithms achieved detection rate of an average of 83% of the phishing sites. The authors proposed that with use of appropriate pre-processing, the detection accuracy would improve.

Enhanced Dynamic Rule Induction (eDRI), is one of the first Covering algorithms that has been applied as an anti-phishing tool (Thabtah, Qabajeh & Chiclana, 2016). This Covering algorithm processes datasets by using two main thresholds, frequency and Rule strength. eDRI scans the training dataset and only stores "strong" features if their frequency exceeds the minimum frequency threshold. As a result, all these features become part of the rule while all other values are removed during the initial scan. Once a rule is derived, eDRI removes its training instances and updates the strong features frequency to reflect the removal of its instances. Hence, eDRI somehow naturally prunes features and leads to a more controllable models. As part of the experiments, 11,000 websites were collected from multiple sources to evaluate eDRI's reliability. In comparison decision tree algorithm, the results acquired showed eDRI superiority to other Covering and decision tree approaches with respect to phishing detection rate.

A machine learning technique that has been highly criticised as a result of its time consumption in tuning its parameters is trial and error Neural Networks (Mohammad, Thabtah & McCluskey, 2013). This technique usually requires a domain expert available during the parameter tuning stage. A Neural Network anti-phishing model proposed the elimination of trial and error and aimed for a more self-structuring classification (Thabtah, Mohammad & McCluskey, 2016). The authors designed the self-structured approach by updating several parameters, like the learning rate dynamically before adding a new neuron to the hidden layer. Therefore, the process of updating the NN features is performed while building the classifier in the network environment. The purpose of applying the dynamic NN model was to detect phishing instances from a real

dataset found in the UCI data repository using different epoch sizes (100, 200, 500, 1000) (Mohammad, Thabtah & McCluskey, 2015). The results revealed promising predictions when compared to Bayesian networks and decision trees.

Phishers keep on updating their deceptive methods hence there was need to develop an anti-phishing NN model that is based on constantly improving the learnt predictive model based on previous training experiences (Mohammad, Thabtah & McCluskey, 2014). The goal was to cope with the aggressive efforts by phishers that frequently update deceptive methods, therefore developed a self-structuring NN classification algorithm that deals with vitality of phishing features. This self-structuring NN algorithm uses validation data to keep track on the performance of the built network model and involves appropriate intelligent decisions based on the outcomes acquired against the validation set. For instance, when the attained error against the network is less than the minimum achieved error so far, the algorithm saves the networks' weights and continues the training process. On the other hand, when the achieved error is larger than the minimum achieved error so far, the algorithm continues the training process without saving the weights. Moreover, if need be, updates on other important network parameters occur during the construction of the classifier without having to wait until the model has been entirely built. As part of the experimentation on a number of features dataset revealed that the self-structuring NN model was able to generate highly predictive anti-phishing models compared to traditional classification approaches, such as C4.5 and probabilistic approaches.

2.3 *Training and testing the model*

The Validation and Testing modules of the NN model includes two components of "Sample Matrix" and "Output Matrix" as follows.

- "Sample Matrix": this matrix contains sample data from the "Input Matrix". The trained NN model uses the data in the "Sample Matrix" as inputs during the testing phase. In our implementation, this matrix is a logical n x 5 matrix contains n sample data from the

"Input Matrix".

- "Output Matrix": this matrix contains output data for the data in the "Sample Matrix". The trained NN model predicts the output values for the "Sample Matrix" and stores them in the "Output Matrix". In our implementation, this matrix is a logical n x 1 matrix contains output data for the emails represented in "Sample Matrix". The trained NN model predicts the output value, in terms of an email being benign or phishing, for each email in the "Sample Matrix". These predictions will be stored in the "Output Matrix" and will be used to evaluate the performance of the neural network. We used 70% of the entire dataset, which includes all the benign emails from PhishTank dataset and all the phishing emails from PhishCorpus dataset, for training, 15% for validation and 15% for testing. We used scikit learn framework to develop, train, validate and then test our classifier. model. Our developed NN model has 10 hidden layers, 5 input features, 1 output layer, and 1 output features, Figure 3. The captured results are discussed in the next section.

Scikit-learn *KFold* class will be used to automatically implement k-fold cross-validation on the given data set.

3. Methodology

3.1 *Introduction*

Machine learning is made up of training phase and the testing phases. We intent to use two datasets to train our model; benign and phishing emails.

We will obtain the datasets sets from Alexa for Benign emails and PhishTank for phishing emails. We intend to use a combination of two algorithms;

1. Random Forest;
2. AdaBoost; to increase accuracy of RF.

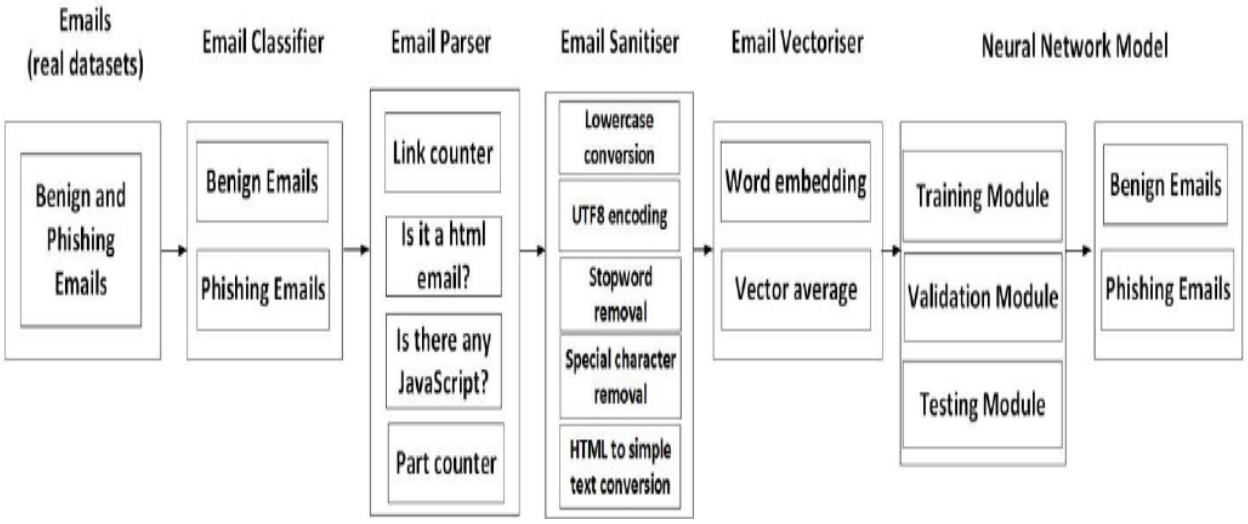

Figure 1. ML classifier modelling of the proposal

Tools to assist in data modeling

- ❑ Open source threat data training set called EMBER.
- ❑ IBM Watson
- ❑ Existing Anti-Phishing solutions like Spam Assassin.
- ❑ PhishTank and Alexa for data sets

Programming languages

- ❑ Python using libraries like *scikit-learn, pandas, numpy and matplotlib*
- ❑ Java

To train and test our classier, we will use a method called 10-fold cross validation. In this method, the training dataset is divided into 10 parts; 9 of the 10 parts will be used to train our classifier and the information obtained from the training phase will be used to validate (test) the 10^{th} part. This process is repeated 10 times in such a way that at the end of the training and testing processes each of the parts will be used as both training and testing data. The cross validation technique ensures that training data id different from test data. In the area of machine learning, this method has shown to provide very good estimate of error of classifier.

3.2 *Model features*

Features used in the email classification

This section describes the phishing features that our classifier will use. These features are identified from different literature thus forming a combination set of features that effectively classify phishing and non-phishing emails.

This project will to use a total of 15 features identified from different literature commonly used by phishing attackers. These features are described below.

IP-based URLs

Legitimate websites normally contain the name of the website on the URL for example http://www.forexample.com/, tells the user one wants to connect to the website of forexample. Attackers usually mask their identity by replacing the domain name with IP address, e.g., http://42.56.100.21/login.asp. By doing so the attackers are able to evade detection by using IP-based URLs which is an indication of a potential phishing attack. This feature is identified in the literature (Fette, Sadeh & Tomasic, 2007).

"HREF" attribute and LINK Text mismatch

A link to another website is usually defined by use of html <a> anchor tag. "href" attribute allows a user to visit another website by describing the location of the second website to be visited. The link is rendered to the browser after a user clicks the "link text" (e.g., a href="URL Address">Link text). Link text could be a plain text, image or any element. If there is a match between the link text and the pointed website, then the website could be a phishing website. All the emails are checked for mismatch between the link text and the href attribute recording a positive Boolean feature if found.

Availability of "Link," "Click," and "Here" in Link Text of a Link

Links in most phishing emails contain certain words like "Click"," Here", "Login", or "Update". All emails are checked for presence of these words and a Boolean value is recorded based on the presence or absence of these words.

Dot contained in domain name

According to Emigh (2007) ("Phishing attacks: Information flow and chokepoints"), the number of dots that should be contained in a legitimate domain name should not exceed three. If the number of dots in URL exceeds three, then a binary value of 1 is recorded to assist in phishing features.

HTML email

Every email is defined by MIME standards which defines the types of components contained in the email. The component content type which is defined by content-type attribute could be plain text denoted by "text/plain", HTML denoted by "text/html". According to Fette et al., an email is a potential phishing email if it contains "text/html" attribute. They argued that is hard to achieve phishing attack without using HTML links.

Use of JavaScript

JavaScript is a scripting language that is used to perform a particular action. This is accomplished by either embedding in the body of an email using <script > tags or in a link using anchor <a> tag. Some malicious actors use JavaScript to evade detection by hiding information from users with the use of JavaScript. Fette et al. if an email is found to contain a JavaScript code in either the body of the email or in a link then it is classified as potential phishing email.

Links found in an email

The number of links contained in an email is recorded and used as feature to detect phishing emails. According to (Yuan & Zhang, 2012), phishing emails normally contain multiple links to malicious websites thus multiple links should be used as a phishing detection feature.

Domain Names in an email

This refers to the number unique domain names extracted from all the referenced URLs. The occurrences are recorded and the value is used as feature. Each occurring domain name is counted once and any subsequent occurrence is discarded. It is believed if an email contains multiple domain names then it is a potential phishing email.

Body_From Domain Match

All the domain names contained in an email are extracted which are then matched with the sender's domain name. The senders' domain name if extracted from the "From" field of the email. If there is a mismatch between the comparison, this could be a potential phishing email.

Word List

Phishing emails usually contain some occurring words which can be used as phishing detection features. These words will be categorized into six different categories whereby each category will be used as a single detection feature. This translates to having six different phishing features. In each category, every word is counted and duplicates discarded (normalized). These categories are:

i. Confirm; Update
ii. Customer; Client; User
iii. Restrict; Hold; Supesnd
iv. Account; Verify; Notification
v. Password; Click; Login; Username
vi. Social Security; SSN

3.3 *Training, testing and validation*

3.3.1 *Training module*

The Training module includes three components of: "Input Matrix", "Target Matrix", and "Fitness Network" as follows.

• "Input Matrix": this matrix contains all the benign emails from Spamcorpus dataset and all the phishing emails from PhishCorpus dataset that the NN model uses in training stage. These emails have been already: parsed by the "Email Parser", sanitised by the "Email Sanitiser", and vectorised by the "Email Vectoriser", Figure.1. In our implementation, this matrix is a logical *14,370 x 5* matrix which represents a matrix with *14,370* rows and *5* columns. *14,370* represent the total number of the emails in our implementation, which is *6,656* for benign emails and *7,714* for phishing emails precisely. 5 represents the size of the assigned vectors to the emails which carries five features for each email: the number of links in the email body, whether or not the email is an HTML email, whether or not there is JavaScript in the email, the number of the email's parts, and the vector average.

• "Target Matrix": this matrix includes all the decisions (benign or phishing) for all the emails.

These decisions are for each and every email stored in the "Input Matrix". In our implementation, this matrix is a logical *14,370 x 1* matrix where *14,370* represent the total number of the emails while *1* represents the size of the assigned decision vector to each email which either carries 0 (benign) or 1(phishing) as a value.

• "Fitness Network": this is the NN model with *n* layers with *x* inputs and *y* outputs where the data from 'Input' and 'Target' matrixes are used for training, validation, and testing, respectively. In our implementation, our NN model has 10 hidden nodes or 10 layers/neurons where 70% of the data from 'Input' and 'Target' matrices are used for training, 15% for validation, and 15% for testing.

3.3.2 *Validation and Testing modules*

The Validation and Testing modules of the NN model includes two components of "Sample Matrix" and "Output Matrix" as follows.

• "Sample Matrix": this matrix contains sample data from the "Input Matrix". The trained NN model uses the data in the "Sample Matrix" as inputs during the testing phase. In our implementation, this matrix is a logical n x 5 matrix contains n sample data from the

"Input Matrix".

• "Output Matrix": this matrix contains output data for the data in the "Sample Matrix". The trained NN model predicts the output values for the "Sample Matrix" and stores them in the "Output Matrix". In our implementation, this matrix is a logical n x 1 matrix contains output data for the emails represented in "Sample Matrix". The trained NN model predicts the output value, in terms of an email being benign or phishing, for each email in the "Sample Matrix". These predictions will be stored in the "Output Matrix" and will be used to evaluate the performance of the neural network. We used 70% of the entire dataset, which includes all the benign emails from PhishTank dataset and all the phishing emails from PhishCorpus dataset, for training, 15% for validation and 15% for testing. We used scikit learn framework to develop, train, validate and then test our classifier. model. Our developed NN model has 10 hidden layers, 5 input features, 1 output layer, and 1 output features, Figure 3. The captured results are discussed in the next section.

S+3656+cikit-learn *KFold* class will be used to automatically implement k-fold cross-validation on the given data set.

3.3.3 *Data source*

The data will be collected from two online sources, one for benign URLs and the other one for phishing URLs.

The benign URL dataset will be collected from Alexa; which is a free open source data repository site that ranks URLS based on their popularity and non-malicious. The phishing email will be retrieved from the PhishTank website which is a free community website that allows users globally to submit, verify, track and share phishing URL data (PhishTank, 2016).

The online datasets will be both cleansed by removing any duplicates and for the experiments both a training and testing set will be created. The training set consists of 4000 URLs, 3000 from the benign set and 1000 from the malicious set. The testing set consists of 7000 URLs, 3000 from the benign set and 4000 from the malicious set. All URLs were selected randomly, except any URLs selected in the testing set do not include those that were present in the training set.

The next step is to extract features from the URLs. To ensure quality between features, all numeric values will be normalised such that their values lie between 0 and 1. All features in the below table are counts and binary values of specific entities within the URL.

Acknowledgements

This research did not receive any specific grant from funding agencies in the public commercial, or not-for-profit sectors.

The authors declare no competing interests.

References

Abdehamid, N. (2015). Multi-label rules for phishing classification. *Applied Computing and Informatics*, Vol. 11 (1), 29-46.

Abdelhamid, N., Thabtah, F., & Ayesh, A. (2014). Phishing detection based associative classification data mining. *Expert systems with Applications Journal*, *41*(2014) 5948-5959.

Abdelhamid, N., & Thabtah, F. (2014). Associative Classification Approaches: Review and Comparison. *Journal of Information and Knowledge Management* (JIKM), *13*(3).

Aburrous, M., Hossain, M., Dahal, K. P., & Thabtah, F. (2010). Experimental case studies for investigating e-banking phishing techniques and attack strategies. *Journal of Cognitive Computation*, *2*(3), 242-253.

Afroz, S., & Greenstadt, R. (2011). PhishZoo: Detecting phishing websites by looking at them. In *Fifth International Conference on Semantic Computing* (18-21 September 2011). Palo Alto, California USA, IEEE.

Akinyelu, A. A., & Adewumi, A. O. (2014). Classification of phishing email using random forest machine learning technique. *Journal of Applied Mathematics*, vol. 2014, Article ID 425731, 6 pages.

Altaher, A., Wan, T. C., & ALmomani, A. (2012). Evolving fuzzy neural network for phishing emails detection. *Journal of Computer Science*, *8*(7).

APWG Phishing Attack Trends Reports (2018). https://www.antiphishing.org/resources/apwg-reports/.

Basnet, R., Mukkamala, S., & Sung, A. H. (2008). Detection of phishing attacks: A machine learning approach. In *Soft Computing Applications Industry* (pp. 373-383). Berlin: Springer.

Behdad, M., T. French, M. Bennamoun, & L. Barone (2012). Nature-inspired techniques in the context of fraud detection. In *IEEE Transactions on Systems, Man, and Cybernetics C.*

Bouckaert, R. (2004). Bayesian network classifiers in Weka. In *Working paper series*. University of Waikato, Department of Computer Science. No. 14/2004. Hamilton, New Zealand.

Bright, M. (2011) Miller Smiles [Online] Available at: http://www.millersmiles.co.uk/ [Accessed 9 January 2016]. *Computer Engineering, and Applied Computing*, pp. 682-686.

Brown, S., Ofoghi, B., Ma, L., & Watters, P. (2017). Detecting phishing emails using hybrid features. In *Symposia and workshops on ubiquitous, autonomic and trusted computing* (UIC-ATC '17), IEEE, Australia.

Cranor, L. F., J. I. Hong, & Y. Zhang (2016). Cantina: A content-based approach to detecting phishing web sites. In *16th International World Wide Web Conference* (WWW '07), Canada.

Cutler, A., & Breiman, L., (2007). *Random forests-classification description*. Department of Statistics Homepage.

Emigh, A. (2007). Phishing attacks: Information flow and chokepoints. In *Phishing and countermeasures: Understanding the increasing problem of electronic identity theft*, USA.

Fette, I., Sadeh, N., & Tomasic, A. (2007). Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web* (pp. 649-656).

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1), 119-139.

Gaines, B. R., & Compton, J. P. (1995). Induction of ripple-down rules applied to modelling large databases. *Intell. Inf. Syst.*, *5*(3), 211-228.

Gupta, M., P. Prakash, R. R. Kompella, & M. Kumar (2015). PhishNet: Predictive blacklisting to detect phishing attacks. In *IEEE Conference on Computer Communications*.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, *11*(1).

Han, W., Cao, Y., & Le, Y. (2015). Anti-phishing based on automated individual white-list. In *4th ACM Workshop on Digital Identity Management* (DIM) (pp. 51-59). ACM USA.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, *11*, 63-90.

Huber, M., Mulazzani, M., Leithner, M., Schrittwieser, S., Wondracek, G., & Weippl, E. (2011). Computer security applications. In *27th Annual Computer Security Applications Conference*.

Khonji, M., A. Jones, & Y. Iraqi (2013). Phishing detection: A literature survey. *IEEE Communications & Surveys Tutorials*.

Ledesma, R., Chou, N., Mitchell, J. C., & Teraguchi, Y. (2014). Client-side defence against web-based identity theft. In *11th Annual Network & Distributed System Security Symposium*, USA.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill, New York, NY, USA.

Mohammad, R., Thabtah F., & McCluskey L. (2015B). Phishing websites dataset. Available: https://archive.ics.uci.edu/ml/datasets/Phishing. Accessed: January 2016.

Mohammad, R., Thabtah F., & McCluskey L., (2014A). Predicting phishing websites based on self-structuring neural network. *Journal of Neural Computing and Applications*, *25*(2), 443-458.

Mohammad, R., Thabtah F., & McCluskey L., (2015A). Tutorial and critical analysis of phishing websites methods. *Computer Science Review Journal*, *17*, 1-24.

Mohammad, R., Thabtah F., & McCluskey, L., (2014B). Intelligent rule based phishing websites classification. *Journal of Information Security* (2), 1-17. ISSN 17518709. IET.

Mohammad, R. M., Thabtah, F. & McCluskey, L. (2013). Predicting phishing websites using neural network trained with back-propagation. In *World Congress in Computer Science, Computer Engineering, and Applied Computing* (pp. 682-686). Las Vigas.

Nargundkar, S., Tiruthani, N., & Yu, W. D. (2017). PhishCatch—a phishing detection tool. In *33rd Annual IEEE International Computer Software and Applications Conference* (COMPSAC '17), USA.

Nazif, M., B. Ryner, & C. Whittaker (2010). Large-scale automatic classification of phishing pages. In *17th Annual Network & Distributed System Security Symposium* (NDSS '10). The Internet Society, USA.

Platt J. (1998). Fast training of SVM using sequential optimization. In *Advances in kernel methods support vector learning* (pp. 185-208). MIT Press, Cambridge.

Qabajeh I., Thabtah, F., & Chiclana, F. (2015). Dynamic classification rules data mining method. *Journal of Management Analytics*, *2*(3), 233-253.

Quinlan, J. (1993). *Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Sadeh, N., Fette, I., & Tomasic, A. (2017). Learning to detect phishing emails. In *16th International World Wide Web Conference* (WWW '17), Canada.

Smadi, S., Aslam, N., Zhang, L., Alasem, R., & Hossain, M. A. (2015). Detection of phishing emails using data mining algorithms. *Computer and Information Sciences*, 1-8.

Strobel, S., Glahn, S., Moens, M. F., & Bergholz, A. (2010). New filtering approaches for phishing email. *Journal of Computer Security*, *18*(1), 7-35.

Tan, C. L., Chiew, K. L., & Sze, S. N. (2017). Phishing webpage detection using weighted URL tokens for identity keywords retrieval. In Ibrahim, H., Iqbal, S., Teoh, S., & Mustaffa, M. (Eds.), *9th International Conference on Robotic, Vision, Signal Processing and Power Applications*. Lecture Notes in Electrical Engineering, vol 398. Springer, Singapore.

Thabtah F., Mohammad R., & McCluskey L. (2016B). A dynamic self-structuring neural network model to combat phishing. In the *Proceedings of the 2016 IEEE World Congress on Computational Intelligence*. Vancouver, Canada.

Thabtah F., Qabajeh I., & Chiclana F. (2016A). Constrained dynamic rule induction learning. *Expert Systems with Applications*, *63*, 74-85.

Wattenhofer, R., Burri, N., & Albrecht, K. (2015). Spamato-an extendable spam filter system. In *Proceedings of the 2nd Conference on Email and Anti-Spam* (CEAS '15), USA.

Witten, I. H., & Frank E. (2005). *Data mining: Practical machine learning tools and techniques*. Morgan Kaufmann Publishers.

Yuan, Y., & Zhang, N. (2012). Phishing detection using neural network. http://cs229.stanford.edu/proj2012/ZhangYuan-PhishingDetectionUsingNeuralNetwork.pdf.

Zhang, Y., Cranor, L. F., Hong, J. I, & Egelman, S. (2016). Phin ding phish: An evaluation of anti-phishing toolbars. In *14th Annual Network & Distributed System Security Symposium*, USA.


C O A S