Predicting Phishing Websites based on Self-Structuring Neural Network

Rami M. Mohammad
School of Computing and Engineering
University of Huddersfield
Huddersfield, UK.
rami.mohammad@hud.ac.uk

Fadi Thabtah
School of MIS
Philadelphia University
Amman, Jordan.
fadi@cud.ac.ae

Lee McCluskey
School of Computing and Engineering
University of Huddersfield
Huddersfield, UK.
t.l.mccluskey@hud.ac.uk

Abstract — Internet has become an essential component of our everyday social and financial activities. Nevertheless, internet-users may be vulnerable to different types of web-threats which 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 is considered as a form of web-threats that is defined as the art of impersonating a website of an honest enterprise aiming to obtain confidential information such as usernames, passwords and social security number. So far, there is no single solution that can capture every phishing attack. In this article, we proposed an intelligent model for predicting phishing attacks based on Artificial Neural Network "ANN" particularly self-structuring neural networks. Phishing is a continuous problem where features significant in determining the type of webpages are constantly changing. Thus, we need to constantly improve the network structure in order to cope with these changes. Our model solves this problem by automating the process of structuring the network and shows high acceptance for noisy data, fault tolerance and high prediction accuracy. Several experiments were conducted in our research, the number of epochs differs in each experiment. From the results, we find that all produced structures have high generalization ability.

Keywords- Web Threat, Phishing, Information Security, Neural Network, Data Mining.

1. INTRODUCTION

Internet is not only important for individual users but also for organizations doing business online, these organizations normally offer online trading [1]. Nevertheless, 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. Therefore, internet suitability for commercial transactions becomes doubtful. Phishing is considered a form of web-threats that is defined as the art of impersonating a website of an honest enterprise aiming to acquire private information such as usernames, password's and social security numbers [2]. Phishing websites are created by dishonest persons to impersonate webpages of genuine websites. These websites have high visual similarities to the legitimate ones in an attempt to defraud the honest internet-users. Social engineering and technical tricks are commonly combined together in order to start a phishing attack [2]. Phishing websites have become a serious problem not only because of the increased number of these websites but also the smart strategies used to design such websites, therefore even users having good experience in the computer security and internet might be deceived. Typically, phishing attack starts by sending an e-mail that seems to be from an authentic organisation to victims urging them to update or validate their information by following a URL link within the e-mail. E-mails have remained the main spreading channel for phishing links since 65% of phishing attacks start by visiting a link received within an e-mail [3]. Other methods of distributing phishing URLs include, Black Hat search engine optimization (Black Hat SEO) [4], Peer-to-peer file sharing, vulnerable websites such as blogs, forums, instant messaging (IM), Internet Relay Chat (IRC), etc.

There are many ways to combat phishing, among them:

- *Legal solutions:* Followed by many countries, the United States was the first to enact laws against phishing activities and many phishers have been arrested and sued. Phishing has been added to the computer crime list for the first time on January 2004 by "Federal Trade Commission" "FTC" which is a U.S government agency aims to promote consumer protection. In March 2005, the "Anti-Phishing Act" was introduced in the U.S Congress by senator "Patrick Leahy". In 2006, the UK government strengthened its legal arsenal against fraud by prohibiting the development of phishing websites and enacted penalties of up to 10 years. In 2005, the Australian government signed a partnership with Microsoft to teach the law enforcement officials how to combat different cybercrimes. Nevertheless, criminal act does a fragile job of preventing phishing attacks since it is very difficult to trace phishers. Moreover, phishing attacks can be performed quickly, later the phisher may disappear into cyberspace and thus the law enforcement authorities must quickly respond because on average the phishing website lives for only 54 hours [5].

- *Education*: The key principle in combating phishing and information security threats is consumer's education. If internet-users could be convinced to inspect the security indicators within the website then the problem is simply gone away. However, the most important advantage for phishers to successfully con internet-users is that most internet-users lack basic knowledge of current online threats that may target them and how the online sites are formally contacting their consumers in case of maintenance and information update issues. In addition, users may ignore checking the security-indicators within the website such as the existence of the SSL protocol, since they are focused on their main tasks, while paying attention to security indicators is considered secondary task [6]. Moreover, some users do not know what SSL-protocol and some other security indicators mean. Generally speaking, although education is an effective technique; getting rid of phishing by teaching would be hard that is since users are required to spend a long time learning phishing methods, and phishers becoming more talented in mimicking legitimate websites and creating new phishing techniques; which makes security experts sometimes deceived.

- *Technical solution:* Weaknesses that appeared when relying on previously mentioned solutions led to the emergence need to innovative solutions. Several academic studies, commercial and non-commercial solutions are offered these days to handle phishing. Moreover, some non-profit organizations such as "APWG", "PhishTank" and "MillerSmiles" provide forums of opinions as well as distribution of the best practices that can be organized against phishing. Furthermore, some security enterprises; for example "MacAfee" and "Symantec" offered several commercial anti-phishing solutions. The success of anti-phishing techniques mainly depend on recognizing phishing websites accurately and within an acceptable timescale. Although a wide variety of anti-phishing solutions are offered, most of these solutions were unable to make decisions perfectly on whether the website is phishy or not, causing the rise of false positive decisions, which means labelling legitimate site as phishing..

Hereunder we preview the most popular approaches in designing technical anti-phishing solutions.

- *Blacklist Approach*: Where the requested URL is compared with a predefined phishing URLs. The downside of this approach is that the blacklist usually cannot cover all phishing websites since a newly created fraudulent website takes considerable time before it is being added to the list. This gap in time between launching and adding the suspicious website to the list may be enough for the phishers to achieve their goals. Hence, the detection process should be extremely quick, usually once the phishing website uploaded and before the user starts submitting his credentials.

- *Heuristic Approach*: The second technique is known as heuristic-based approaches, where several features are collected from the website to classify it as either phishy or legitimate. In contrast to the blacklist method, a heuristic-based solution can recognize freshly created phishing websites in real-time [7]. The effectiveness of the heuristic-based methods, sometimes called features-based methods, depends on picking a set of discriminative features that could help in distinguishing the type of website [8].

2. MOTIVATION

Phishing websites are expected to be more stylish in the future. Therefore, a promising solution that must be improved constantly needed to keep pace with this continuous evolution. As internet-users feel safe of being phished if they utilize anti-phishing tools. This throws a great obligation on the anti-phishing tools to be accurate in predicting phishing websites. Predicting and stopping fraudulent websites is a critical step toward protecting online transactions. Several approaches were proposed to discover and prevent these attacks and as we mentioned earlier anti-phishing measures may take several forms including legal, education and technical solutions. The technical solutions are the subject of our interest, particularly, heuristic-based phishing detection approach. The accuracy of the heuristic-based solution mainly depends on a set of discriminative criteria's extracted from the website. Hence, the way in which those features are processed plays an extensive role in classifying websites correctly. Therefore, an effective and fast knowledge retrieval method is essential for making a good decision. Data mining is one of the techniques that can make use of the features extracted from the websites to find patterns as well as relations among them [9]. Data mining is important for decision-making since decisions may be made based on the patterns and rules achieved by the data-mining algorithm. Although plenty of applications offered for combating phishing websites, few of them make use of data mining techniques in distinguishing phishing websites from legitimate ones. Besides, most of these suggested methods are inapplicable, inaccurate and produce an improper level of false positive rates [10]. Phishing detection problem is a type of classification task. The classification task goal is to assign each test data to one of the predefined classes. Phishing is considered a binary classification problem since the target class has two possible values "Phishy" or "Legitimate". Once a webpage is loaded on the browser a set of features will be extracted from it. Those features have a strong influence in determining the type of the webpage. An example of such features includes "IP address, long URL, uses '@', https and SSL, age of domain, etc". Those features will be stored in a data storage called vector. A data-mining model then will process the data in vector, make

some calculations and finally classify the webpage to either "Phishy" or "Legitimate".
Classification in data mining is commonly used to solve classification problems; which is learning from historical data patterns in order to classify new data accurately. Phishing detection falls within the scope of this type.
However, when data mining applications are spoken about these days most likely people are talking about either decision trees or neural networks.
Neural network "NN" is a well-known classification technique. NN is a model of the human brains and nervous system, since human brain and the neural network are composed of interconnected processing units called neurons [11]. The link that connects neurons to each other has a value that signifies the relative importance of each input to a neuron and it is called connections weights [11] that are the crucial elements in any neural network model. Connections weights are adjusted repeatedly during the training phase until reaching an acceptable solution. A trained neural network is considered as an expert in the field of information to which it is applied.

The neural network is an example of non-linear prediction methods that has been used in many domains like pattern recognition, speech recognition, handwriting recognition, biological classification and documents classification. Neural network proved its superiority for many reasons, among them:

- Nonlinearity: NN is an effective technique in modelling classification problems where the output values are not directly related to its input.
- Adaptive: NN has the ability to adjust the weights based on the changes of its surrounding environments.
- Generalisation: NN is able to find the correct output for the unseen inputs.
- Fault-tolerance: NN performance does not significantly affected under difficult circumstances such as losing the connection between some neurons, noisy or missing data.
- Identical designing steps: The same principles, scheme and methodological steps are employed in designing ANN in all domains [12].

The motivation behind our study is to build a robust and effective model based on neural network to detect phishing websites on the fly.
In this article, we try to answer the following research questions:

1- The applicability of neural network in predicting phishing websites.
2- What is the best neural network architecture for predicting phishing websites?
3- How neural network can be trained to achieve a high predictive performance.

This article structured as follows: Section III discusses related works and highlights different phishing detection methods presented in the literature. Section IV describes the features used in our model. Section V, introduces traditional neural network modelling techniques. Section VI details the description of our model. Sections VII conduct several experiments and we conclude in Section XI.

3. RELATED WORK

Although several solutions were offered to tackle phishing, most of these solutions are not capable to make a decision perfectly thus increasing the false positive rate. In this section, we review current intelligent anti-phishing approaches as well as the techniques they utilize in developing solutions.
One approach employed in [13]is based on experimentally contrasting associative classification algorithms. The authors have gathered 27 different features from various websites as shown in Table 1. Those features ranged among three fuzzy set values "Legitimate, Genuine and Doubtful". To evaluate the selected features, the authors conducted experiments using the following data mining techniques, MCAR [14], CBA [15], C4.5 [16], PRISM [17], PART [9] and JRip [9]. The results showed an important relation between "Domain Identity" and "URL" features. There was insignificant impact of the "Page Style" on "Social Human Factor criteria".
Later on [18], the authors used the 27 features to build a model to predict websites type based on fuzzy data mining. Although, their method is a promising solution it did not clarify how the features were extracted from the website and specifically features related to human factors "Much Emphasis on Security and Response, Generic Salutation and Buying Time to Access Accounts". Furthermore, their model works on multilayered approach i.e. each layer should have its own rules; however, it was not clear if the rules were established based on human experience, which is one of the problems we aim to resolve in this article, or extracted in an automated manner. Moreover, the authors classified the website as very-legitimate, legitimate, suspicious, phishy or very-phishy, but they did not clarify what is the fine line that separate one class from another. Generally, fuzzy data mining uses approximations; that does not make good candidates for managing systems that require extreme precision [19].

Another method proposed in [20] suggested a way to detect phishing websites by capturing abnormal behaviours demonstrated by these websites. Two components used as phishing detectors:

1. The identity extractor: This is the organization's full name abbreviation along with a unique string shown in the domain name.

2. Page classifier: Some web properties i.e. structural features that are relevant to the site identity cannot be fabricated.

Table 1 E-BANKING PHISHING CRITERIA

Features Group	Phishing Factor Indicator
URL & Domain Identity	Using IP Address
	Request URL
	URL of Anchor
	DNS Record
	Abnormal URL
Security & Encryption	SSL Certificate
	Certification Authority
	Abnormal Cookie
	Distinguished Names Certificate(DN)
Source Code & Java script	Redirect Pages
	Straddling Attack
	Pharming Attack
	Using onMouseOver
	Server Form Handler
Page Style & Contents	Spelling Errors
	Copying Website
	"Submit" Button
	Using Pop-Ups Windows
	Disabling Right-Click
Web Address Bar	Long URL Address
	Replacing Similar Characters for URL
	Adding Prefix or Suffix
	Using the @ Symbol to Confuse
	Using Hexadecimal Character Codes
Social Human Factor	Much Emphasis on Security and Response
	Generic Salutation
	Buying Time to Access Accounts

Structured website consists of "W3C DOM" features [21]. The authors have selected six structural features:
(Abnormal URL, abnormal DNS record, abnormal anchors, Server form handler, abnormal cookies and abnormal certificate in SSL). Support-Vector-Machine classifier "Vapnik's" [22] was used to determine whether the website is phishy or not. Experiments on a dataset consist of 279 phishing websites and 100 legitimate websites showed that the "Identity Extractor" performs better in dealing with phishing pages because the legitimate websites are independent from each other, whereas some of the phishing sites are correlated. Moreover, "The Page Classifier" performance mainly depends on the result extracted from "Identity Extractor". The classification accuracy in this method was 84%, which is relatively considered low. However, this method snubs important features that can play a key role in determining the legitimacy of the website, which explains the low detection rate. One solution to improve this method could be by using additional features such as security related features.

The method proposed in [10] suggested utilizing "CANTINA" which is a content-based technique to detect phishing websites using the term-frequency-inverse-document-frequency (TF-IDF) measures [23]. CANTINA stands for "Carnegie Mellon Anti-phishing and Network Analysis Tool" and it checks the webpage content then decides whether it is phishing or not by using TF-IDF. TF-IDF produces weights that assess the word importance to a document, by counting its frequency. CANTINA works as follows:

1. Calculate the TF-IDF for a given webpage.
2. Take the five highest TF-IDF terms and add them to the URL to find the lexical signature.
3. The lexical signature is fed into a search engine.

If the N tops search results having the current webpage, it is considered a legitimate webpage. If not, it is a phishing webpage. N was set to 30 in the experiments. If the search engine returns zero result, thus the website is labelled as phishy, this point was the main drawback of using such technique since this would increase the false positive. To overcome this weakness, the authors combined TF-IDF with some other features those are:
(Age of domain, known images, suspicious URL, IP address, dotes in URL and Forms).
A limitation of this classification method is that some legitimate websites consist mostly of images so using the TF-IDF may not be right. In addition, this approach does not deal with hidden texts which might be effective in detecting the type of the webpage.

Another approach that utilizes CANTINA with an additional attributes proposed in [24]. The authors have used 100 phishy websites and 100 legitimate ones, which are considered limited in their experiments. According to CANTINA, there are eight features have been used for detecting phishing websites (domain age, known image, suspicious URL, suspicious link, IP address, dots in URL, Forms and TF-IDF"). Some changes to the features have been performed during the experiments as follow:

1. The "Forms" feature is considered as a filter to start the process of decision-making about the legitimacy of the website since fraud websites that may cause users' information to be lost must contain "Forms" with input blocks.
2. The "Known image" and "Domain age" features are ignored since they are insignificant according to the authors.
3. A new feature that shows the similarity between doubtful webpage and top-page of its domain is suggested.

The authors have performed three types of experiments against their dataset where the first one evaluated a reduced CANTINA feature set "dots in URL, IP address, suspicious URL and suspicious link", and the second experiment involved testing whether the new features "domain top-page similarity" are significant enough to play a key role in detecting website type. The third experiment evaluated the results after adding the new suggested feature to the reduced CANTINA features utilized in

the first experiment. By comparing the newly model performance after adding the new feature the results of all compared classification algorithms showed that the new feature played a key role in detecting the type of the website. The best accurate algorithm was neural network with an error rate equals to 7.5%, followed by SVM and random-forest with an error rate equals to 8.5%, and daboost with 9.0% and J48 with 10.5%, whereas Naïve Bayes gave the worst result with a 22.5 % error rate.

In [25], the authors compared a number of commonly used machine-learning methods including SVM, rule-based techniques, decision trees, and Bayesian techniques. A random forest algorithm was implemented in "PILFER". PILFER stands for (Phishing Identification by Learning on Features of email Received) which essentially aim to detect phishing emails. A dataset consisting of 860 phishing emails and 6950 legitimate emails was used in the experiments. The proposed technique correctly detected 96% of the phishing emails with a false positive rate of 0.1%. The authors used 10 features for detecting phishing email's those are:
"IP based URL's, age of domain, non-matching URL's, having a link within the e-mail, HTML emails, number of links within the e-mail, number of domains appears within the e-mail, number of dot's within the links, containing JavaScript and spam filter output"

Table 2 Features added to PILFER to classify websites

Phishing Factor Indicator	Feature Clarification
Site in browser history	If a site not in the history list then it is expected to be phishing.
Redirected site	Forwarding users to new webpage.
tf-idf (term frequency-inverse document frequency)	Searching for the key terms on a page and checking whether the current page is present in the result.

PILFER can be applied towards classifying websites by combining all the 10 features except "Spam filter output" with those shown in Table II. For assessment; the authors utilized exactly the same dataset in both PILFER and SpamAssassin version 3.1.0 [26]. One other goal of using SpamAssassin was actually to extract "Spam filter output" feature. The results revealed that PILFER has a false positive rate of 0.0022% if it is being installed without a spam filter. If PILFER is joined with SpamAssassin the false positive rate decreased to 0.0013%, and the detection accuracy rises to 99.5%.

One promising approach proposed by [27] detected type of websites based on visual similarity by comparing phishing websites with the legitimate ones. This technique initially decomposed the webpage into salient block regions depending on "visual cues." The visual similarity between phishing webpage and legitimate one is then evaluated using three metrics: block level similarity; layout similarity, and overall style similarity based on the matching of the salient block regions. A webpage is considered phishy if any metric has a value higher than a predefined threshold. The authors collected 8 phishing webpages and 320 official bank pages and they conducted their experiment which shows a 100% true positive and 1.25% false positive. Although the results were impressive, this work suffers from subsequent weaknesses:

1. The dataset size was relatively considered very low.
2. Potential instability attributed to the high flexibility of the layout within the HTML documents.

In [28], a new method, called "Dynamic Security Skins" was disseminated. Since both; system designers and phishers rely on user interface to protect or deceive users; this approach used a shared secret image that allows a remote server to prove its identity to the user in a way that supports easy verification by users. This technique requires the users to make verification based on comparing the user expected image with an image generated by the server. The authors implement their schema by developing an extension to "Mozilla Firefox browser". The main disadvantage of this schema is that the users bear the burden of deciding whether the website is phishing or not, thus users need to be conscious of the phishing and look for signs that the website he is visiting is in fact a spoof website. This approach also suggests a fundamental change in the web infrastructure for both servers and clients, so it can succeed only if the whole industry's support it. In addition, this technique does not provide security if the users logged-in from a public computers.

In 2010, a survey presented in [7] aimed to evaluate the performance of machine-learning-based-detection-methods including: "AdaBoost, Bagging, SVM, Classification and Regression Trees, Logistic Regression, Random Forests, NN, Naive Bayes and Bayesian Additive Regression Trees" showed that 7 out of 9 of machine-learning-based-detection-methods outperformed CANTINA in predicting phishing websites those are:
"AdaBoost, Bagging, Logistic Regression, Random Forests, Neural Networks, Naive Bayes and Bayesian Additive Regression Trees". A dataset consisting of 1500 phishing websites and 1500 legitimate websites used in the experiments. The evaluation based on eight heuristics presented in CANTINA. A set of pre-experiments decision was taken as follows:

- The number of trees in Random Forest is set to 300.
- For all experiments need to be analysed iteratively the iteration time was set to 500.
- Threshold value was set to zero for some machine-learning techniques such as Bayesian Additive Regression Trees (BART).

- Radial based function was used in SVM.
- The number of hidden neurons was set to five in the NN experiments.

4. PHISHING WEBSITES FEATURES

There are several features distinguish phishing websites from legitimate ones. In our study we used 17 features taking either a binary or a ternary value. Binary value features hold either "Phishy" or "Legitimate" since the existence or lack of the feature within the website determines the value assigned to that feature. Whereas for ternary value features one more value has been added this is "Suspicious". For ternary value features, the existence of the feature in a specific ratio determines the value assigned to that feature. The features used in our study were explained below.

1. Using IP address: Using IP address in the hostname part of the URL address means users can almost be sure someone is trying to steal his personal information. This feature is a binary feature.

An example of using IP address is as follows:

http://91.121.10.211/~chems/webscr/verify

Sometimes the IP address is transformed to hexadecimal form as follows:

http://0x58.0xCC.0xCA.0x62

2. Long URL: Phishers resort to hide the suspicious part of the URL, which may redirect the information submitted by the users or redirect the uploaded page to a suspicious domain. Scientifically, there is no reliable length distinguishes phishing URLs from legitimate ones. As in [29], the proposed length of the legitimate URLs is 75. However, the authors did not justify the reason behind this value. In our previous article [30] we find that if the URL length is less than 54 characters then the URL is classified as "Legitimate", and if the URL length ranges from 54 to 75 the website is classified as "Suspicious", otherwise the website is classified as "Phishing". This feature is a ternary feature.

3. URLs having "@" symbol: As we stated earlier, phishers attempt to hide the suspicious part of the URL. One of the things that cause suspicion is the existence of the "@" symbol in the URL. However, the "@" symbol leads the browser to ignore everything prior the "@" symbol and redirect the user to the link typed after it. This feature is a binary feature.

4. Adding Prefixes and Suffixes to URL: Phishers try to deceive users by reshaping the URL to look like the legitimate ones. A technique used to do so is by adding prefix or suffix to the legitimate URL thus the user may not notice any difference. This feature is a binary feature.

5. Sub-domain(s) in URL: Another technique used by the phishers to deceive the users is by adding sub-domain(s) to the URL thus the users may believe that they are dealing with a credited website. As we mentioned in our previous article [30] this feature is a ternary feature that is since the URL address is considered "Suspicious" if it has one sub-domain, and considered "Phishy" if the sub-domains within the URL is more than one. Whereas, for the URLs that do not have sub-domains "Legitimate" value will be assigned.

6. Misuse of HTTPs: The existence of the HTTPs every time sensitive information is being transferred reveals that the user certainly connected with an honest website. However, phishers may use fake HTTPs so that the users may be deceived. In our previous article [30] we recommended to check if the HTTPs is offered by a trusted issuer such as "GeoTrust, GoDaddy, Network Solutions, Thawte, and VeriSign". For this feature, if the HTTPs exists but the certificate issuer is not within the trusted issuer list we will assign "Suspicious". Whereas, if the HTTPs is not existing at all we will assign "Phishy". Otherwise, we will assign "Legitimate". This feature is a ternary feature.

7. Request URL: A webpage consists of a text and some objects such as images and videos. Typically, these objects are loaded on the webpage from the same domain where the webpage exists. If the objects are loaded from a domain different from the domain typed in the URL address bar then the webpage is potentially compromised a phishing suspicion. The ratio of the objects loaded from a different domain identifies the value assigned to this feature. In our previous article [30] if the ratio is less than 20% then this website is considered "Legitimate", but if the ratio ranges between 20% to 50% then this website is considered "Suspicious", otherwise the website is considered "Phishy". This feature is a ternary feature.

8. URL of Anchor: An anchor is an element defined by the <a> tag. This feature is treated exactly as "Request URL" but for this feature the links within the webpage might refer to a domain different from the domain typed on the URL address bar. This feature is a ternary feature and treated exactly as "Request URL".

9. Server Form Handler "SFH": Once the user submits his information, that information will be transferred to a server to be processed. Normally, the information is processed from the same domain where the webpage is being loaded. Phishers resort to make the server form handler either empty or the submitted information is transferred to somewhere different from the legitimate domain. As we mentioned in our previous article [30] there are three possible cases for this feature those are:

- The SFH is empty and then we will assign “Phishy”.
- The SFH refers to a different domain and then we will assign “Suspicious”.
- The SFH is associated to the same domain shown in the address bar and then we will assign “Legitimate”.

10. Abnormal URL: If the website identity does not match its record shown in the WHOIS database [31] then the website is classified as “Phishy”. This feature is a binary feature.

11. Redirect Page: This feature is commonly used by phishers by hiding the real link and ask the users to submit their information to a suspicious website. Nevertheless, some legitimate websites may redirect the user to a new website to submit his credentials. The fine line that distinguishes the phishing websites from the legitimate ones is the number of redirect pages used within the website. As we mentioned in our previous article [30] if a website is redirected less than 2 times then the website is classified as “Legitimate”, but if the website is redirected 2,3 or 4 times then the website is considered “Suspicious”, and if the website is redirected more than 4 times then the website is considered “Phishy”.

12. Using Pop-up Window: It is unusual to find a legitimate website asks users to submit their credentials through a popup window, this feature is a binary, since if the website asks the users to submit their credentials through a popup window we will assign “Phishy” otherwise we will assign “Legitimate”.

13. Hiding the Suspicious Links: Phishers resort to hide the suspicious link by showing a fake link on the status bar of the browser or by hiding the status bar itself. This can be achieved by tracking the mouse cursor and once the user arrives to the suspicious link the status bar content is changed. This feature is a binary feature since if the website code contains “onMouseOver” and the code assigned to that event cause the URL shown on the status bar to be changed then we will assign “Phishy” otherwise we will assign “Legitimate”.

14. DNS Record: If the DNS record is empty or not found then the website is classified as “Phishy”, otherwise it is classified as “Legitimate”. Phishers aim to acquire sensitive information as fast as possible, that is since the phishing website lasts for a short period of time and then the URL is not valid any more. DNS record provides information about the domain that is still alive, while the deleted domains are not available on the DNS record. This feature is a binary feature.

15. Website Traffic: Legitimate websites are usually having high web traffic since they are visited regularly. Phishing websites having a relatively short life thus their web traffic is either not exists or their web traffic rank is less than the limit that gives it the legitimate status. In our previous article [30] we assigned “Legitimate” for the websites ranked among the top 100,000 websites, and we assigned “Suspicious” for the websites ranked more than 100,000. If the website has no traffic record or not being recognized by Alexa database we will assign “Phishy”. This feature is a ternary feature.

16. Age of Domain: For this feature and as we stated in our previous article [30] the website is considered “Legitimate” if the domain aged more than 2 years. However, if the domain age is less than 2 years and more than 1 year we will assign “Suspicious”. Otherwise, the website is considered “Phishy”. This feature is a ternary feature.

17. Disabling Right Click: Phishers use JavaScript to disable the right click function so that users cannot view and save the source code. As we stated in our previous article [30] this feature is not commonly used by phishers since it appeared only 40 times on a dataset consist of 2500 instances. However, the website is classified as “Phishy” if the right click is disabled. Otherwise, the website is classified as “Legitimate”. This feature is a binary feature.

5. TRADITIONAL MODELLING OF NEURAL NETWORKS

In this section, we explain what NN is and we review a set of concepts related to it.
The main objective of this study is to automate the process of developing a neural network model that can be used to predict phishing attacks. A number of sub-goals have been identified towered this end, those are:

- Collecting the dataset patterns that will be used in our experiments and pre-process them into a form that is suitable for training neural networks.
- Determine the neural network architecture as well as the learning rate that will yield the best predictive performance.
- Show that neural networks can be used as a valid and effective approach to predict phishing websites.

Although there are several definitions of neural networks, they all agreed on that the neural network model consists of a set of simple processing units called neurons and a set of weighted connections between these neurons. These weighted connections are repeatedly adjusted during training of the network until reaching a suitable solution. How the neurons are connected and the strength of these connections defines the behaviour of the neural network. The following steps describe the overall tasks involved in constructing a neural network model.

A. Data Collection and Preperation

Our 17 features presented in section IV were used to represent the input neurons. A dataset consists of 1400 phishing and legitimate websites were used to extract the 17 features using our own tool [30] [32]. The dataset composed of 600-legitimate website collected from yahoo directory [33] and starting point directory [34], and 800-phishing website collected from Phishtank archive [35] and Millersmiles archive [36]. The collected dataset holds categorical values those are "Legitimate", "Suspicious" and "Phishy", these values should be converted to numerical values, so that the neural network can do its calculations thus we will replace the values 1,0 and -1 instead of "Legitimate", "Suspicious" and "Phishy" respectively.

B. Network Architecture

This includes the types of connections within the network, the order of the connections and the values of the weights.

One class of neural network architectures is the feed-forward networks. For this class, the data always propagate in unidirectional form starting from the input layer down to the output layer.

The other class of neural network architecture is the recurrent neural network, which contains feedback connections from units in the subsequent layers to units in the preceding layers. Recurrent networks have feedback connections between neurons of different layers or loop type self-connections. This implies that the output of the network not only depends on the external inputs, but also on the state of the network in the previous training iteration. Determining the network architecture is one of the difficult tasks in constructing any model but one of the most essential steps to be taken. The neural network architecture employed in this study is feed-forward with one hidden layer, which sometimes called multi-layered perceptron. The advantage of multi-layered perceptron is that the number of neurons in the hidden layer can be changed to adapt to the complication of the relationships between input and output variables. Although neural network construction has been widely researched, there is no known procedure or algorithm for the general case. However, one of the experimental objectives of this study was to conclude the size of the hidden layer that produces the best predictive performance.

C. Network topology

The topology of a network is specified by the number of layers, number of neurons in every layer and the weighted connections among all neurons. These types of layers are the input, hidden and output layer.

In feed-forward network, data always propagates in one way from input layer to output layer passing through the hidden layer(s) if any. The input layer receives input data from external world and a neuron in this layer is called an input neuron. In the network architecture, the input neurons symbolize the data presented to the network for processing. In our model the 17 features shown in section IV represent the input neurons, whereas, the website visited by the user represent the external world from which these features are extracted.

The layer following the input layer is the hidden layer, and neurons in this layer are called hidden neurons. The hidden layer receives inputs from the previous layer, transforms those inputs into nonlinear combinations and passes the results to the next layer for further processing. The hidden layer can consist of one or more layers of neurons. Commonly, the networks with one hidden layer are used in modelling since it has been found that more than one hidden layer does not produce a major improvement in the neural network performance [11]. Moreover, using more than one hidden layer makes the neural network computationally complex. In our model, we used only one layer of hidden neurons while the number of neurons within this layer was changeable.

Two approaches have been proposed in specifying the number of neurons in the hidden layer those are:

- Pruning: By starting with a large number of neurons, and then progressively some of these neurons removed during training until the desired performance is met.
- Constructive: By starting with a small number of neurons, and then increase the number of neurons during training until the performance of the network reaches an acceptable level.

The constructive approach was adopted in this study since this method is more suitable to our problem and was shown to be more successful [37].

The output layer is the final layer of the network, and the neurons in this layer are called output neurons. The neurons in this layer represent the output of the network.

The network size must be considered when constructing a network that is since the smaller network size requires fewer storage and have higher processing during training but such network sometimes contains several local minima [38]. Larger networks have a tendency to learn fast in term of training iterations required and have increased ability to avoid local minima, but they need a large number of training samples in order to reach better generalisation ability [39].

D. Network Parameters

The main goal of training a network is to adjust its weight vector. The step size taken to adjust the weights during the training is a function of a set of network parameters.

The network parameters include "learning rate, momentum value, error function, epoch size and transfer functions".

Normally, preparing the network parameters starts by initializing the weights. In our model, the weight adjustment is achieved by an error-correction learning rule called the delta rule or "Widrow-Hoff learning rule" as shown in Equation 1.

$$\Delta W(i) = \eta . err(i) . x(i) \quad (1)$$

Where "*ΔW*" is the weight-adjustments value for the "*i-th"* input variable. "err" is the error value and *"x"* is the input value. "η" is a constant value specified by the user defines the learning rate. The learning rate plays a very important role in the learning process, since it controls the speed at which the neural network finds the optimal solution. However, if the learning rate value is very big then the learning will be very fast but with the risk that the network will diverge from the solution. On the other hand, a small value learning rate means that the network will take a very long time to converge to the final solution. The delta rule can be modified by adding a momentum term as shown in Equation 2 to increase the convergence of the model without affecting the network stability, where *"α"* denotes the momentum value, and *ΔW(i - 1)* is the weight-adjustment value during the previous adjustment step. Typically, the momentum value is set to a positive value ranged between 0.1 and 1 [40].

$$\Delta W(i) = \eta . err(i) . x(i) + \alpha . \Delta W(i-1) \quad (2)$$

After calculating the adjustment weight, we find the new weight as follow:

New weight= old weight + adjustment weight.

An important parameter that is commonly taken into consideration in neural network is the error function, which is the function that is to be improved during training. In our study, the mean square error "*MSE*" is used because it is calculated easily and because it is penalise large errors. The mean square error is calculated based on Equation 3:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(predicted\ value_i - desired\ value_i)^2 \quad (3)$$

Where "N" is the total number of training examples, "$predicted\ value_i$" Is the value produced by the network for training-example "I" and "$desired\ value_i$" Is the actual value.

E. Training the Network

A correct mapping of input to output requires determining the correct weights for the neural network. Optimizing the connection weights is identified by training or learning the network. The network learns by adapting the strength of its connection weights by examining the training patterns presented to it based on a specific training algorithm. The main goal of training the neural network is to reduce the error in the network output by adjusting the weight vector. Two learning approaches can be used to learn the neural networks namely, supervised approach and un-supervised approach. In supervised learning approach, a set of training examples is given along with the desired output of each example. While in un-supervised approach, training examples are supplied without any information about the desired output. Supervised learning approach is hence used in application where a desired output is known and where the network performance can be assessed by comparing the network outputs with the desired output. For phishing detection, supervised approach is used since the desired output is provided with each training example.

Back-Propagation algorithm is adopted in our study to adjust the network weights. The back-propagation algorithm is described as the following pseudocode:

Initialize the weights vector
S = the training set fed to the network
Repeat
    For each "input-output" pair denoted by P in S
        In = input pattern in P
        Out = desired output
        Compute network output (netout)
        network error = Out – netout
    End For
  Find weight change for weights connecting hidden to output
  Find weight change for weights connecting input to hidden
  Update weights
Until reaching (a satisfactory network error value OR maximum iteration)

6. PREDICTING PHISHING BASED ON SELF-STRUCTURING NEURAL NETWORK

As we mentioned earlier, one of the difficult tasks associated with building a neural network model is that it is necessary to specify the network architecture in terms of the number of hidden layers and the number of neurons in each hidden layer. In addition, a set of parameters (learning rate, momentum, epoch size) should be specified in advance in order to build a good model. Unfortunately, it is hard to identify in advance the appropriate network structure for a particular application, and that could be reached by trial and error.

A neural network that is structured incorrectly may produce an under-fitted model. On the other hand, exaggeration in restructuring the system to suit every item in the training dataset may cause the system to be overfitted. For overfitted models, the error value of the training dataset is small, but when new data fed to the model, the error is big. One possible solution to the overfitting problem is by adding new neurons to the hidden layer, or sometimes adding a new layer to the network. Overfitting caused by the noisy data, which occurs whenever there are irrelevant features presented within the training dataset. However, acquire a noisy free dataset is a difficult task, and so, an acceptable error margin should be specified while building the model. Which itself considered a problem, since the user may not be able to determine the acceptable error rate. Sometimes the user specifies the acceptable error rate to a value that is un-reachable, or even specifies a value that can be

improved. For traditional data mining algorithms (C4.5, CBA, PART … etc.) the user is not asked to specify the acceptable error rate. Moreover, the phishing problem is a continuous problem, that means; new features having a strong influence in determining the website type are expected to appear in the future or even some currently used features may no longer effective in predicting the type of the website. Thus, we need to improve the network structure constantly to cope with these changes. Our model solves this problem by automating the process of structuring the network.

One downside of using neural network is that it is difficult to interpret its results and it is regarded as a black box. However, we believe that the difficulty in interpreting the results will add a positive edge to our model since, as the phisher has the ability to design and manage a phishing website; he might have good skills in hacking the anti-phishing tool and interpret its content; and thus he can circumvent it. Moreover, most users are not interested in interpreting the neural network results, all what they care about is a way protecting them from phishing.

Our model shown in Fig 1 will address the aforementioned problems; the most important characteristics of our model can be summarized as follows:

1- Self-structuring: The model will search for the most appropriate structure in terms of the number of hidden neurons and the learning rate value.
2- Minimal number of parameters: In our model, the model-designer is asked to provide the dataset and the maximum number of epochs only, while in traditional neural network modelling technique the model-designer must specify too many parameters. Moreover, in our model the model-designer is not involved in specifying the acceptable error since the model will search for a structure providing the minimum error rate.
3- Adaptable: As we stated earlier, the features used in predicting the type of a website might be changed, thus designing a fixed neural network structure means that some of the currently used features could be no longer effective in classifying the website. However, since our model is self-structuring model then the model-designer have just to collect a new dataset periodically and fed it to the model, thus the new result will be produced.
4- The model could be installed on a dedicated server, and a tool, which is integrated with a web-browser, may contact this server frequently to obtain updates if any.

The model works as follow:

Step 1. At the beginning, the model creates the simplest neural network structure, which consists of only one neuron in the hidden layer. Whereas the number of neurons in the input and output layers; is determined based on the problem at hand. In our case the number of neurons at the input and output layers is set to 17 and 1 respectively. Small non-zero random values will be assigned for each connection weight.

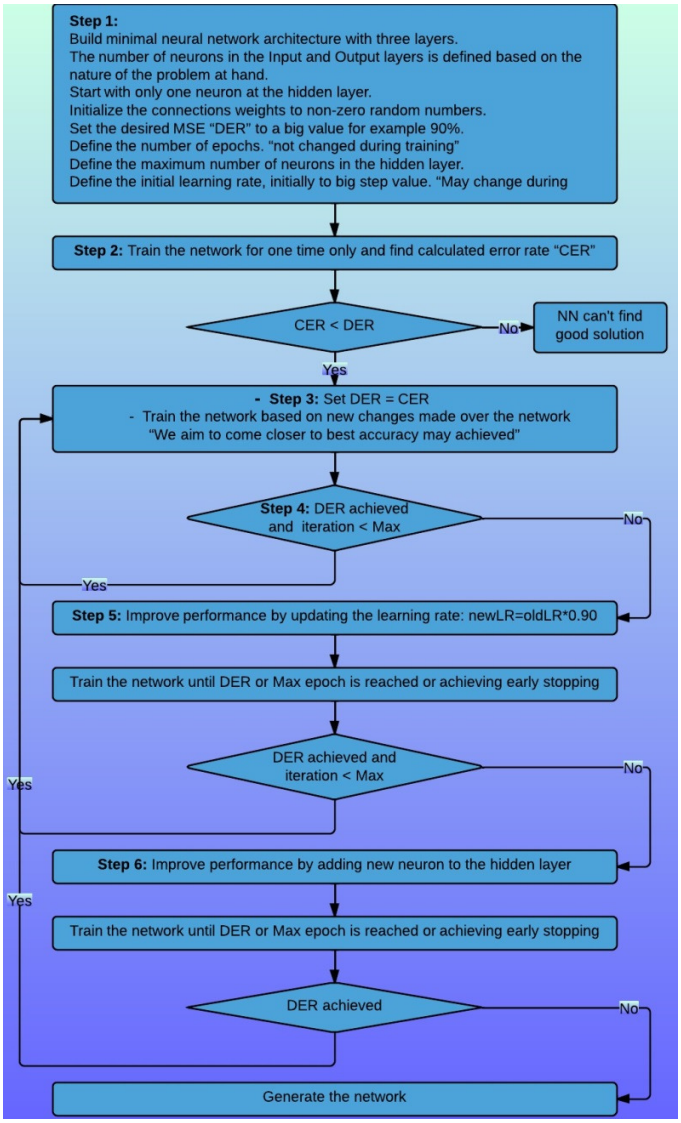

Figure 1 Self-structuring NN model

We assumed that the learning rate is set to a big value aiming to converge quickly to the possible solutions. Hence, this value will be adjusted during the network training. For traditional neural network modelling techniques, the learning rate is set to a fixed value that is not changed during the training phase. In our model, we ran quickly to possible solutions and then by adjusting the learning rate we slow-down and examine all possible solutions more deeply. In addition, we assumed the initial learning rate is 0.8, and the initial desired-error-rate is set to a big value; we assume it 90%.The model-designer must specify the maximum number of epochs.

Step 2. In this step, the model will find the calculated error rate "CER". The model will run one epoch only aiming to determine what the desired error rate "DER" to be achieved in the next iteration(s) is.

Step 3. Train the network until the "DER" or maximum number of epochs is achieved or achieving early stopping.

Step 4. If "DER" is achieved before reaching the maximum epochs, this could be an indication that the current structure and current learning rate may be able to improve the network accuracy in the next iteration(s), thus we set DER = CER and go back to step 3. Else, we go to step 5.

Step 5. If the maximum number of epochs is reached without achieving the "DER", we maintain the network structure and try to improve the network accuracy by adjusting the learning rate. Unlike other constructive-neural networks, our model attempts to find the optimal solution as well as the simplest structure. The traditional constructive neural networks attempt to improve the network accuracy by adding new neuron to the hidden-layer or add a new hidden layer and ignore adjusting the learning rate. Our model leaves the network expansion as a last option. However, the main reason of adjusting the learning rate is that in some regions of the weight-space, the gradient is large and we need a large step size; that is why we start with a high learning rate value. Whereas, in other regions, the gradient is small and we need a small step size, this happens whenever we come closer to a local minimum. We assumed to adjust the learning rate by decrease it 10% as shown in Equation 4.

$$\eta' = \eta * 0.90 \qquad (4)$$

After adjusting the learning rate we set DER = CER and train the network. If DER is achieved then we go back to step 3 aiming to improve the network performance based on the new learning rate. Else, we go to step 6.

Step 6. If we cannot achieve the "DER" in step 5, then we assume that the network ability of processing information is insufficient therefore, the model will add a new neuron to the hidden layer and train the network. If adding new neuron improved the network accuracy then we go to step 3 aiming to update the DER or the learning rate before deciding to add new neuron. Else, if adding new neuron to the network does not improve the network accuracy, then the training process will terminated and the final network will be generated.

7. EXPERIMENTS

A. *Experimental Methodology*

An object oriented C++ program was created to implement our model. All experiments were conducted in a system with CPU Pentium Intel® Core™ i5-2430M @ 2.40 GHz, RAM 4.00 GB. The environment is Windows 7 64-bit Operating System. The dataset composed of 600-legitimate website and 800-phishing website was collected. We are interested in obtaining a model with optimal generalisation performance. However, most NN models are criticized being overfitting the input data, which means, while the error rate on the training dataset decreases during the training phase, the error rate on the unseen dataset (testing dataset) increases at some point. To overcome this problem, we used the "Hold-Out" validation technique, by dividing our dataset into training, validation and testing datasets. The examples in each dataset were selected randomly. After training, we ran the network on the testing dataset. Error on the testing dataset offers an unbiased approximation of the generalization error. We split our dataset to 20% for testing and 80% for training. Then the training dataset is divided to 20% for validation and 80% for training. Another way to avoid overfitting is to stop training as soon as the error on the validation dataset starts to increase. However, the validation dataset may have many local minima, thus if we stop training at the first increase we may lose some points that achieve better results because the error rate may decrease again at some points. Therefore, we track the validation error, and if the current error is less than the previously achieved error then we update the weights and keep training the network. On the other hand, if the currently achieved error is bigger than the previously achieved error we do not update the weights and keep training until the fraction between the current error and the smallest error exceeds a certain threshold, in our model the threshold is assumed to 20%. Formula (1) clarifies how the early stopping is handled in our model.

Where, ω_j is the currently achieved error, and ω_i is the minimum error.

$$IF \begin{cases} \omega_j < \omega_i \xrightarrow{weights\ updated} Keep\ training \\ (\omega_j > \omega_i) and (\omega_j < 1.2 * \omega_i) \xrightarrow{weights\ not\ updated} Keep\ training \\ Otherwize \rightarrow Training\ terminated \end{cases}$$

Formula 1

In our model, "Log-sigmoid" activation function was used for all layers. The momentum value was assumed to 0.7, and the initial learning rate was assumed 0.8. However, one of the experimental goals is to determine the learning rate value that produces the best predictive performance. The initial weights were initialized to random values ranging from -0.5 and +0.5. The maximum number of possible neurons in the hidden layer is set to 8.

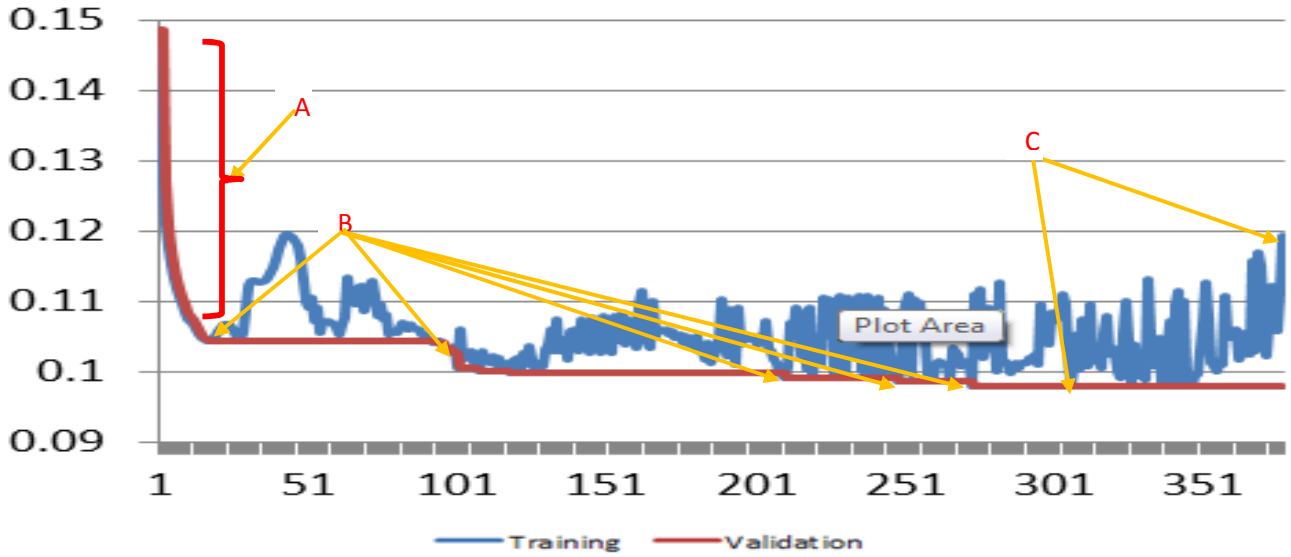

Figure 2 Evolution of the Training Error

A. *Experimental Results*

Several experiments were conducted; in each experiment, we changed the number of epochs. From the results shown in Table 3, it is clear that our model was able to design NN with acceptable generalization ability. For instance, the results obtained when the number of epochs = 500 showed that the prediction accuracy of the testing dataset was close to the accuracy achieved from training and validation datasets.

This means while the error decreased on the training dataset it is also decreased on testing dataset.

Table 3 Experimental Results

Epochs	Optimal number of HN	Training set Accuracy	Validation set Accuracy	Testing set Accuracy	MSE	Best Learning Rate
50	4	91.32%	90.03%	90.35%	0.0629	0.7684
100	4	92.33%	90.84%	91.35%	0.0453	0.7308
200	4	93.07%	91.23%	91.80%	0.0922	0.6609
500	3	93.45%	91.12%	92.48%	0.0280	0.5799
1000	3	94.07%	91.31%	92.18%	0.0248	0.5799

Fig 2 shows the evolution of the training error when the epoch number equals 500.

From the Fig 2, a set of important observations may be summed up as follow:

- At point “A”, it was clear that the gradient is large while at other points when approaching the generalization state the gradient is small. That is why we started with a large learning rate and adjust it during training.
- At point “B”, the error on the validation dataset becomes smaller, thus the model will save the weights at these points and keep training hoping to find better points. At other points, the weights are not saved because the error rate did not improve.
- At point “C”, the fraction between the minimum and the maximum error rate exceeded our threshold thus the model stopped training and it will try to improve the network performance either by adjusting the learning rate or by adding new neuron(s) or even terminate the training and produce the network.

A. *A Practical Example on Predicting Website Class*

In this section, we will explain how the websites are classified using our NN model. Suppose that the features extracted from a webpage are shown in Table 4 where the values "1", "0" and "-1" denote "Legitimate", "Suspicious" and "Phishy" respectively.

Table 4 Practical Example

Feature	Value
Using IP address	1
Long URL	0
URL having @ Symbol	0
Adding Prefix and Suffix	1
Sub-Domain(s)	1
Misuse of HTTPs	0
Request URL	1
URL of Anchor	-1
Server Form Handler	1
Abnormal URL	1
Redirect Page	-1
Using Pop-up Window	-1
Hiding Suspicious Link	0
DNS record	1
Website Traffic	1
Age of Domain	0
Disabling Right Click	1

The final structure produced when the number of epochs is set to 500 is shown in Fig 6. In addition, the weights produced are shown in Table 7 and Table 8.

The first step is by finding the net-input for each hidden neuron by multiplying each input by its corresponding weight. The results are shown in Table 5.

Table 5 Net input for each neuron in the hidden layer

	Hidden Neuron # 1	Hidden Neuron # 2	Hidden Neuron # 3
Net Input	-10.208376	-33.011972	7.476011

Each net-input is passed to the activation function, which is in our model the Log-sigmoid activation function. The result produced is shown in Table 6.

Table 6 Results of Log-Sigmoid activation function

	Hidden Neuron # 1	Hidden Neuron # 2	Hidden Neuron # 3
Log-Sig	0.00004	4.60344^{-15}	0.99943

Then, the net-input is calculated for the output neuron by multiplying the results shown in Table 6 by their corresponding weights shown in Tabe 8 . The result produced "-2.93448056" is passed to the activation function. The final result produced is "0.05048" that is then compared to a predefined threshold, which is in our model "0.5". If the final result > threshold, then the website is classified as legitimate website, otherwise it is classified as a phishy. In our example the result is less than the threshold, thus the webpage is classified as a phishy.

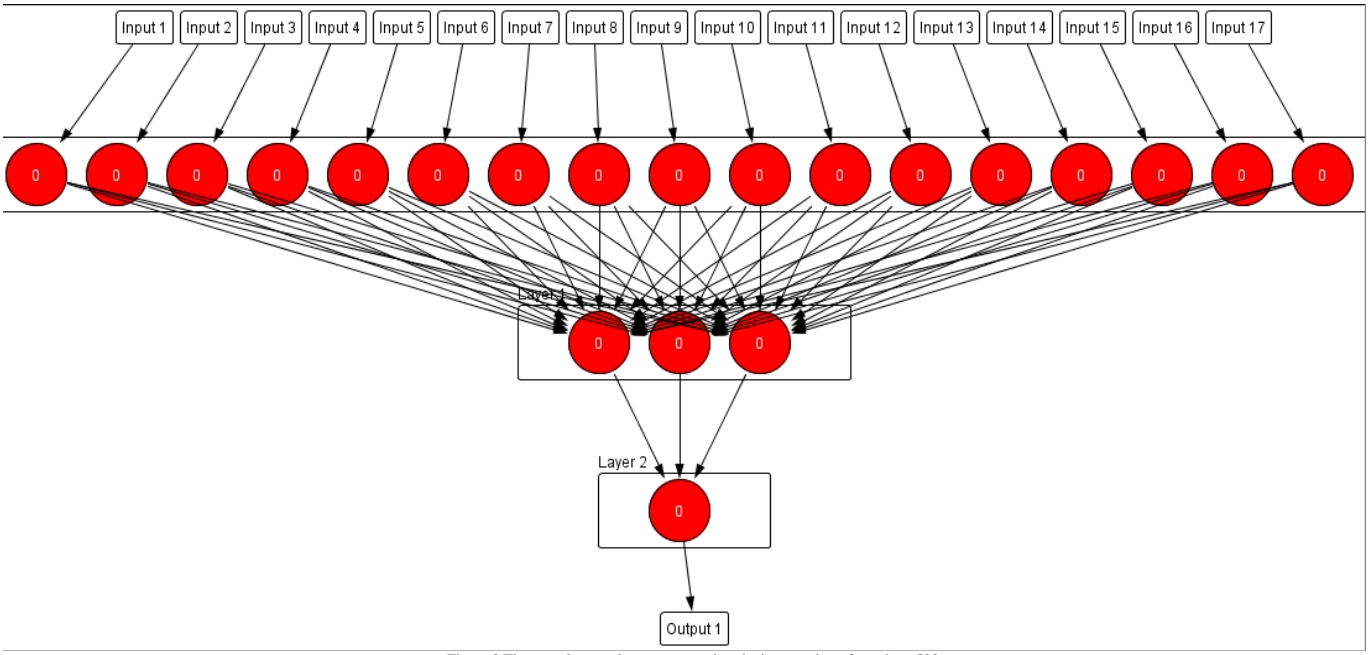

Figure 3 The neural network structure produced when number of epochs = 500

Table 7 Weights produced from input to hidden neurons

Weights produced connecting input neurons to hidden neurons			
Input	**First Hidden Neuron**	**Second Hidden Neuron**	**Third Hidden Neuron**
Using IP address	-2.788467	-1.732674	-10.499482
Long URL	-1.841950	0.657919	-5.551135
URL having @ Symbol	-25.389151	-18.487131	33.564385
Adding Prefix and Suffix	0.059755	6.701862	2.325618
Sub-Domain(s)	-1.638793	0.773444	-0.969623
Misuse of HTTPs	0.765649	3.354878	2.372594
Request URL	-2.053365	9.460433	19.544987
URL of Anchor	0.380302	-2.597401	-2.085049
Server Form Handler	-8.259405	-16.573597	-8.682610
Abnormal URL	11.324954	-19.059105	0.296253
Redirect Page	7.666283	4.066873	-4.492245
Using Pop-up Window	6.681336	17.623121	-1.630921
Hiding Suspicious Link	12.627318	-8.031678	1.087805
DNS record	-1.460111	-3.557257	-1.370880
Website Traffic	-0.202345	7.409626	-2.399567
Age of Domain	-10.442993	3.888161	-0.702606
Disabling Right Click	9.537322	2.657889	1.02310

Table 8 Weights produced from hidden to output

Weights produced from hidden to output layer
19.360754
23.560028
-2.936857

8. CONCLUSION

It is well known that a good anti-phishing tool should predict the phishing attacks in a good time scale. We believe that the availability of a good anti-phishing tool at a good time scale is also important to increase the proportion of predicting phishing websites. This tool should be improved constantly through continuous retraining. Actually, the availability of fresh and up to date training dataset which may acquired using our own tool [30] [32] will help us to retrain our model continuously and handle any changes in the features which are influential in

determining the website class. Although neural network proves its ability to solve a wide variety of classification problems, the process of finding the optimal structure is very difficult and in most cases, this structure is determined by trial and error. Our model solves this problem by automating the process of structuring a neural network scheme therefore if we build an anti-phishing model and for any reasons we need to update it, then our model will facilitate this process. That is since our model will automate the structuring process and will ask for few user-defined parameters. Several experiments conducted in our research, the number of epochs differs in each experiment. From the results, we find that all produced structures have high generalization ability. In addition, results shown in Table 3 revealed that neural network is a good technique in predicting phishing websites. Although the model architecture used in our research seems to be slightly difficult, its principle is the utilization of an adaptive scheme with four mechanisms: structural simplicity, learning rate adaptation, structural design adaptation and early stopping technique based on validation errors. However, there are three major achievements contributing to the better performance of our model:

- The first achievement is that, our model uses an adaptive strategy in designing the network whereas traditional modelling techniques rely on trial and error. In most cases, the trial and error technique consumes time before achieving a network with better generalization ability.
- The second reason is the training method used in our model since we try to improve the network performance as much as possible by adjusting the learning rate before deciding to add a new neuron to the hidden layer.
- The third reason is the generalization ability of our model. Although several algorithms proposed to automate the neural networks design most of them, use a greedy scheme in determining the optimal structure by adding a new layer to the network or adding a new neuron(s) to the hidden layer. The main idea behind our model is to focus on an adaptive scheme for both learning rate and network structure. The adaptive scheme is more convenient because it is able to handle different situations that might be occurred during the designing phase.

One of the future developments of our model is by adding a technique to assess the significance of the features before they are adopted in building a neural network based anti-phishing system. In addition, we are planning to create a toolbar that implements our model and integrate it with a web browser. This toolbar should be updated periodically to cope with any improvements on the weights in case a new model is being created.

REFERENCES

[1] J. Liu and Y. Ye, "Introduction to E-Commerce Agents: Marketplace Solutions, Security Issues, and Supply and Demand," in *E-Commerce Agents, Marketplace Solutions, Security Issues, and Supply and Demand*, London, UK., 2001.

[2] APWG, G. Aaron and R. Manning, "APWG Phishing Reports," APWG, 1 February 2013. [Online]. Available: http://www.antiphishing.org/resources/apwg-reports/. [Accessed 8 February 2013].

[3] Kaspersky Lab, "Spam in January 2012: Love, Politics and Sport," 2013. [Online]. Available: http://www.kaspersky.com/about/news/spam/2012/Spam_in_January_2012_Love_Politics_and_Sport. [Accessed 11 February 2013].

[4] seogod, "Black Hat SEO," SEO Tools, 16 June 2011. [Online]. Available: http://www.seobesttools.com/black-hat-seo/. [Accessed 8 Januery 2013].

[5] R. Dhamija, J. D. Tygar and M. Hearst, "Why Phishing Works.," in *Proceedings of the SIGCHI conference on Human Factors in computing systems*, Cosmopolitan Montréal, Canada, 2006.

[6] L. F. Cranor, "A framework for reasoning about the human in the loop," in *UPSEC'08 Proceedings of the 1st Conference on Usability, Psychology, and Security*, Berkeley, CA, USA, 2008.

[7] D. Miyamoto, H. Hazeyama and Y. Kadobayashi, "An Evaluation of Machine Learning-based Methods for Detection of Phishing Sites," *Australian Journal of Intelligent Information Processing Systems,* pp. 54-63, 2 10 2008.

[8] X. Guang, o. Jason, R. Carolyn P and C. Lorrie, "CANTINA+: A Feature-rich Machine Learning Framework for Detecting Phishing Web Sites," *ACM Transactions on Information and System Security,* pp. 1-28, 09 2011.

[9] I. H. Witten and E. Frank, "Data mining: practical machine learning tools and techniques with Java implementations," ACM, New York, NY, USA, March 2002.

[10] Y. Zhang, J. Hong and L. Cranor, "CANTINA: A Content-Based Approach to Detect Phishing Web Sites," in *Proceedings of the 16th World Wide Web Conference*, Banff, Alberta, Canada, 2007.

[11] B. Widrow, M. and A. Lehr, "30 years of adaptive neural networks," *IEEE press,* vol. 78, no. 6, pp. 1415-1442, 1990.

[12] I. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals, computing, design, and application.," *Journal of Microbiological Methods.,* vol. 43, no. 1, pp. 3-31, 2000.

[13] Aburrous, M, Hossain, M. A., Dahal, K. and Fadi, T, "Predicting Phishing Websites using Classification Mining Techniques," in *Seventh International Conference on Information Technology*, Las Vegas, Nevada, USA, 2010.

[14] F. Thabtah, C. Peter and Y. Peng, "MCAR: Multi-class Classification based on Association Rule," in *The 3rd ACS/IEEE International Conference on Computer Systems and Applications*, 2005.

[15] K. Hu, Y. Lu , L. Zhou and C. Shi, "Integrating Classification and association rule Mining," in *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98, Plenary Presentation)*, New York, USA, 1998.

[16] J. R. Quinlan, "Improved use of continuous attributes in c4.5," *Journal of Artificial Intelligence Research,* pp. 77-90, 1996.

[17] J. Cendrowska, "PRISM: An algorithm for inducing modular rule," *International Journal of Man-Machine Studies,* pp. 349-370, 1987.

[18] M. Aburrous, M. A. Hossain, K. Dahal and F. Thabtah, "Intelligent phishing detection system for e-banking using fuzzy data mining," *Expert Systems with Applications: An International Journal,* pp. 7913-7921, December 2010.

[19] S. A. S., O. S. A. and O. B. A., "Threat Modeling Using Fuzzy Logic Paradigm," *Informing Science: International Journal of an Emerging Transdiscipline.,* vol. 4, no. 1, pp. 53-61, 2007.

[20] Y. Pan and X. Ding, "Anomaly Based Web Phishing Page Detection," in *In ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference.*, Washington, DC, Dec. 2006.

[21] "W3C," [Online]. Available: http://www.w3.org/TR/DOM-Level-2-HTML/. [Accessed December 2011].

[22] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning,* vol. 20, no. 3, pp. 273 - 297, 1995.

[23] C. . D. Manning, P. Raghavan and H. Schütze , Introduction to Information Retrieval, Cambridge University Press, 2008.

[24] Nuttapong Sanglerdsinlapachai and Arnon Rungsawang, "Using Domain Top-page Similarity Feature in Machine Learning-based Web," in *Third International Conference on Knowledge Discovery and Data Mining*, Washington, DC, 2010.

[25] N. Sadeh, A. Tomasic and I. Fette, "Learning to detect phishing emails," *Proceedings of the 16th international conference on World Wide Web,* pp. 649-656, 2007.

[26] T. A. S. Project, "SpamAssassin," [Online]. Available: http://spamassassin.apache.org/. [Accessed January 2012].

[27] L. Wenyin, G. Huang, L. Xiaoyue, Z. Min and X. Deng, "Detection of Phishing Webpages based on Visual Similarity," in *Proceeding WWW '05 Special interest tracks and posters of the 14th international conference on World Wide Web* , New York, NY, USA , 2005.

[28] R. Dhamija and J. D. Tygar, "The battle against phishing: Dynamic Security Skins.," in *Proceedings of the 1st Symposium On Usable Privacy and Security*, New York, NY, USA, July, 2005.

[29] S.-J. Horng, P. Fan, M. K. Khan, . R.-S. Run, J.-L. Lai, R.-J. Chen, A. Sutanto and H. Mingxing, "An efficient phishing webpage detector.," *Expert Systems with Applications: An International Journal.,* vol. 38, no. 10, pp. 12018-12027, 2011.

[30] R. M. Mohammad, F. Thabtah and L. McCluskey, "An Assessment of Features Related to Phishing Websites using an Automated Technique," in *The 7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, London, 2012.

[31] "WhoIS," [Online]. Available: http://who.is/. [Accessed December 2011].

[32] R. M. Mohammad, "Phishing websites Dataset," December 2012. [Online]. Available: http://phishingdatasets.wikispaces.com/. [Accessed December 2012].

[33] "Yahoo Directory," [Online]. Available: http://dir.yahoo.com/. [Accessed December 2011].

[34] "Starting Point Directory," [Online]. Available: http://www.stpt.com/directory/. [Accessed January 2012].

[35] W. Liu, X. Deng, G. Huang and A. Y. Fu, "An Antiphishing Strategy Based on Visual Similarity Assessment," in *IEEE Educational Activities Department Piscataway*, NJ, USA , March 2006 .

[36] "MillerSmiles," [Online]. Available: http://www.millersmiles.co.uk/.

[37] T. M. Nabhan and A. Y. Zomaya , "Toward generating neural network structures for function approximation," *Elsevier Science Ltd.,* vol. 7, no. 1, pp. 88-99, 1994.

[38] R. G. Hutchins, "Neural network topologies and training algorithms in nonlinear system identification," in *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, Monterey, CA., 1995.

[39] Z. M. Jacek, Introduction To Artificial Neural Systems., Jaico Publishing House., 1994.

[40] M. Kantardzic, Data mining: concepts, models, methods, and algorithms, 2, illustrated. ed., John Wiley & Sons., 2011.