An Intelligent Phishing Detection and Protection Scheme using a fusion of Images, Frames and Text

Thesis

Moruf Akin Adebowale

Supervisors:
Professor Alamgir Hossain
Dr. Khin Lwin
Dr. George Wilson

Title

AN INTELLIGENT PHISHING DETECTION AND PROTECTION SCHEME USING A FUSION OF IMAGES, FRAMES AND TEXT

by

MORUF ADEBOWALE

Presented to the Faculty of Science and Technology of

The Anglia Ruskin University at Chelmsford in Fulfilment

of the requirement for the degree of

Doctor of Philosophy

September 2020

Declaration

I certify the following about the thesis entitled “Intelligent Phishing Detection and Protection Scheme Using Image, Frame and Text Fusion” submitted for the degree of “Doctor of Philosophy.”

I am the inventor of all or part of the whole research, which includes content and layout. That where reference is made to the work of others, due acknowledgement is given.

I. The research work(s) are not in any way a violation or infringement of any copyright, trademark, patent, or other rights whatsoever of any person.

II. What if the research work(s) have been supported and commissioned by any organisation, I have fulfilled all the obligations required by such contract or agreement.

I also certify that any material in the thesis which has been acknowledged for a degree by any university or institution is identified in the text.

'I certify that I am the below-named student and that the information provided in the form is

correct.'

Full Name: ……Moruf Adebowale……

Signed: ……  **……**

Date: ……09 September 2020……

Abstract

A phishing attack is one of the most common forms of cybercrime worldwide. In recent years, phishing attacks have continued to escalate in severity, frequency and impact. Globally, the attacks cause billions of dollars of losses each year. Cybercriminals use phishing for various illicit activities such as personal identity theft and fraud, and to perpetrate sophisticated corporate-level attacks against financial institutions, healthcare providers, government agencies and businesses. Several solutions using various methodologies have been proposed in the literature to counter web-phishing threats. This research work adopts a novel strategy to the detection and prevention of website phishing attacks, with a practical implementation through development towards a browser toolbar add-in.

A three-fold approach to the mitigation of phishing attacks is developed. Firstly, a total of 13,000 features and 10,000 images were collected from both phishing and legitimate websites to collate a database that was used in the current work. This database has been donated to the public domain to promote further work on phishing detection within the wider research community. Secondly, a hybrid feature selection approach is adopted. This approach combines the associated elements of images, frames and text of legitimate and non-legitimate websites which can then be collectively processed by an Artificial Intelligence scheme based on the adaptive neuro-fuzzy inference system (ANFIS). Thirdly, an alternative novel approach is evaluated using two deep learning techniques, the Convoluted Neural Network (CNN) and the Long-Short Term Memory (LSTM) variant as a combined classifier called the Intelligent Phishing Detection System (IPDS).

The IPDS is shown to be highly effective both in the detection of phishing attacks and in the identification of fake websites. Experimental results show that an offline approach using the ANFIS has a 98.3% accuracy with an average detection time of 30 seconds, whilst the CNN+LSTM approach has a slightly lower accuracy with an average detection rate of 25 seconds. These times are within typical times for loading a web page which makes toolbar integration into a browser a practical option for website phishing detection in real time. The results of this research are compared with previous work and demonstrates both better or similar detection performance. This is the first work that considers how best to integrate images, text and frames in a hybrid feature-based solution for a phishing detection scheme.

Acknowledgement

First of all, I would like to thank God, ALLAH, whose grace has led me to this critical moment of my life.

I want to thank Professor Alamgir Hossain, Dr Khin Lwin and Dr George Wilson for their direction, assistance, and guidance. Their recommendations and suggestions have been helpful for my research work. Our discussions and meeting have been beneficial and valuable; your comments and advice have helped me a lot to follow the right direction in my research work. Thank you very much for your help and support.

Also, I offer my regards and blessings to my colleagues who helped me in many ways during my research work

Finally, words alone cannot express the thanks and gratitude I owe to my wife Nwanne Adebowale for her encouragement, support and endless love, through the duration of my PhD research. Thanks to my parents for all their prayers and my family and friends for their ongoing support and love.

Table of Contents

List of Figures

List of Tables

List of Charts

List of Acronyms

Acronyms	Description
AIC	Akaike's Information Criterion
ANFIS	Adaptive Neuron-fuzzy Inference System
APWG	Anti- Phishing Working Group
BART	Bayesian Additive Regression Trees
BIC	Bayesian Information Criterion
CART	Classification and Regression Tree
CF	Collaborative Filtering
CSP	Cellular Service Provider
DDoS	Distributed Denial of Service
DNS	Domain Name Server
DOM	Document Object Model
DoS	Denial of Service
FN	False Negative
FIS	Fuzzy Inference System
FP	False Positive
HMM	Hidden Markov Model
HSF	Hybrid Feature Selection
HTML	Hypertext Mark-up Language
IG	Information Gain
IP	Internet Protocol
IPDS	Intelligent Phishing Detection System
ISP	Internet Service Provider
LR	Logistic Regression
MDA	Mail Delivery Agent
MDA	Maximum Dependency Algorithm
MTA	Mail Transport Agent
MUA	Mail User Agent
NNet	Neural Network
RF	Random Forest
SMO	Sequential Minimal Optimisation
SMS	Short Messaging Service
SMTP	Simple Mail Transfer Protocol
SPF	Sender Policy Framework

Acronyms	Description
SVM	Support Vector Machine
TF-IDF	Term Frequency Inverse Document Frequency
TN	True Negative
TP	True Positive
URL	Uniform Resource Locator
WEKA	Waikato Environment for Knowledge Analysis

Published Contributions

The research work carried out as part of the thesis has been published in international scientific refereed conference proceedings, journal and poster presentations.

Conference Papers

I. M. A. Adebowale, K. T. Lwin and M. A. Hossain, "Deep Learning with Convolutional Neural Network and Long Short-Term Memory for Phishing Detection," *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, Island of Ulkulhas, Maldives, 2019, pp. 1-8.

Journal Paper

II. Adebowale, M. A., Lwin, K. T., Sánchez, E. and Hossain, M. A. (2019) 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text', *Expert Systems with Applications,* 115, pp. 300-313.

III. M. A. Adebowale, Lwin, K. T. and Hossain, M. A. (2020) 'Intelligent phishing detection system using deep learning algorithm', *Journal of Enterprise Information Management,*

Presentation

I. 2018 - The School of Computing and Information, Anglia Ruskin University, invited lecture to students in cybersecurity course: 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text', (by Moruf Adebowale, Alamgir Hossain, Khin Lwin, and Erika Sanchez).

II. 2018- The PGR Anglia Ruskin University Student Research Conference presentation 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text' (by Moruf Adebowale, Alamgir Hossain and Khin Lwin).

III. 2017- The Anglia Ruskin University Student Annual PGR Research Conference poster presentation 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text' (by Moruf Adebowale, Alamgir Hossain and Khin Lwin).

IV. 2016- The Anglia Ruskin University Annual PGR Student Research Conference poster presentation 'Intelligent web-phishing detection using adaptive neuro-fuzzy inference system (ANFIS) as a baseline classification algorithm' (by Moruf Adebowale, Alamgir Hossain and Khin Lwin).

V. 2018, 2017 and 2016 – Annual Group Research Seminar Presentation: 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text' (by Moruf Adebowale, Alamgir Hossain and Khin Lwin).

CHAPTER 1: INTRODUCTION

Chapter 1 Introduction

The Internet today has become an effective means of communication with many people using it to generate an online environment to manage offline commercial activities (Arachchilage and Love, 2014). However, even when used to set up a solely online business functionality, despite the benefit that the Internet offers there is also a negative aspect that requires that the user pay attention to issues such as identity theft, fraud, malware and phishing. Phishing is a form of social engineering attack in which an attacker, also known as a phisher, attempts to fraudulently retrieve sensitive user information by sending an email claiming to be a legitimately established organisation. They scam the user into giving confidential information that will be used for identity theft (Arachchilage, Love and Beznosov, 2016). A phisher uses various methods, including email, web pages, and malicious software, to steal personal information and account credentials (Purkait, 2012). The aim of the phishing website is to use users' private information without their permission, and they do this by developing a new website that mimics a reliable website (Upadhyaya, 2012). Hence, phishing website detection has become the object of a great deal of consideration among many academics who are attempting to find ways to incorporate malicious detection devices into web servers as a safety precaution (Hu *et al.*, 2016).

The trend of increasing technology usage is causing the threat of online identity theft to rise (Chang *et al.*, 2013). Also, as individual users primarily use the Internet, it has become crucial that organisations have a presence online. These circumstances have made the security of commercial transactions on the Internet less safe (Yang, Lin and Chen, 2018). As mentioned above, phishing is among the threats that affect web activities, where an attacker mimics the website of an official establishment by gathering the personal data of online users (Imani and Montazer, 2017). Phishing is a complicated problem to tackle as it differs from other types of security threat such as intrusions and malware that are based on the exploiting valnerabilities in the security

of enterprise systems architecture (Moradpoor, Clavie and Buchanan, 2017). Users on the network are the weakest link that phisher targets and the type of users that the phisher decides to attack depend on the users' activities and the attacker's aim in terms of social engineering.

Despite there being several ways to carry out phishing attacks, current phishing detection techniques unfortunately only cover some attack vectors such as fake website and emails (Daeef *et al.*, 2016). Moreover, phishing has become more sophisticated, and such attacks can now bypass the filters that have been put in place by anti-phishing techniques (Hong, 2012). Some detection techniques have been proposed, but most of them only deal with spoof web pages (Tan *et al.*, 2016). However, it is quite challenging in detection due to the evading techniques that the phisher uses.

This chapter reviews the relevant research background which is used to identify the appropriate purpose and scope of this study. Research questions, aims and objectives are subsequently presented, along with a sentence on contribution to knowledge. The chapter concludes with an overview of the thesis structure.

1.1 Background

Web spoofing is the act of luring the user of the Internet to interact with a fake website instead of the original site. A phisher does this by sending an email to a user falsely claiming to be a legitimately established organisation in an attempt to trick the worker into giving sensitive information that will be used for fraudulent activities (Arachchilage, Love and Beznosov, 2016). Cyberattacks, such as phishing and scamming, are anticipated to rise in number and sophistication in the future (Cherdantseva *et al*., 2016).

However, the existing defence mechanisms are not able to stop attacks such as the drive-by download because these attacks are becoming more well organised and

sophisticated (Baykara and Gürel, 2018). Smart detection techniques that can solve the existing phishing problem are therefore required (Shabut, Lwin and Hossain, 2016). Combined techniques that use human factors such as awareness as their basis, as well as a heuristic-based approach, can deliver an active, intelligent-based defence scheme to help users achieve excellent real-time protection for their online transactions (Daeef et al., 2016). Some Internet security products, such as anti-virus software attract a large number of users based on their product features. These tools alert users to the presence of suspicious web pages as part of an anti-theft approach (Shabut, Lwin and Hossain, 2016). Hence, the capability offered by these tools is useful for users who have knowledge of online threats, costs and countermeasures and who can in turn respond to such security warnings (Babu, Nirmala and Kumar, 2010).

A phishing attack may appear in various forms of communication such as messaging, voice over Internet protocol (VOIP), short message service (SMS) and spam emails (Ahmed and Abdullah, 2016). However, phishing attacks are mainly delivered by an email that lures users to click a link in the body of the email that then takes them to an external website that targets their financial information by claiming to be their bank, the inland revenue, a utility company or a government agency (Office for National Statistics, 2017). Financial institutions and end-users are regularly exposed to the threat of phishing attacks (Barraclough *et al.*, 2013). Furthermore, the threat is continuing to grow due to an increase in deception, impersonation, fraud and multiple online attacks (Abbasi *et al*., 2015).

1.2 The Phishing Problem

The phishing scam has evolved in recent years due to productive economic and high-tech conditions worldwide. The rise in all types of fraud loss in 2019 is attributed to the increase in deception scams and impersonation, as well as sophisticated online attacks such as malware and phishing (APWG, 2019). The phisher sometimes claims

that there has been a fraudulent transaction on their target's account that looks suspicious and suggests that they should update or verify their account (Purkait, 2012). There have been several high-profile data breaches (APWG, 2019) reported, as well as some low-level attacks. The data that is acquired can be used to commit fraud directly in the case of card details that can be used for remote purchases and huge amount of money lost (Financial Fraud Action, 2018).

The technical resources needed to execute phishing attacks can be readily acquired through private and public sources (M Jameel and George, 2013). Some technical resources have been streamlined and automated, enabling non-technical criminal individuals to also use them for attacking users online (Mohammad, Thabtah and McCluskey, 2012). Criminal gangs also use malware and phishing emails as a means to compromise customers' details and security. Phishing is one of the most rapidly growing threats to the interconnected world of information technology. It is a system-based attack that exploits human vulnerabilities rather than software vulnerabilities (Mao *et al*., 2017).

Therefore, there is a need to develop a solution that supports the user in identifying a replica website that potentially could be used to host phishing attacks. As phishing attacks pose a severe threat to economy and security globally, there is a strong need for the automation of phishing attack detection using a robust algorithm (Islam and Abawajy, 2013).

Security professionals are seeking to diminish the impact of phishing by filtering spam and phishing emails. Also, educating users and encouraging the use of anti-phishing toolbars that are designed to prevent users from accessing phishing web pages where their sensitive information would be requested and then transmitted to criminals (Liu, Qiu and Wenyin, 2010).

Some tools have also been developed to warn users that the website they are visiting is likely to be fake (Babu, Nirmala and Kumar, 2010). However, these solutions fail to

incorporate various elements of the website such as frame, text and image into a single solution that provide more accurate detection and protection for online user. On this basis this work will take into consideration the website features like image, frame, and text into one system that can prevent the phisher from exploiting the vulnerable who are performing their legitimate activities online. The various methods of detecting phishing web pages can be classified into three types: toolbar based, user-interface based and content-based (Zhang *et al*., 2011).

One approach to preventing the impact of phishing is to terminate the phishing website itself. This action is usually undertaken by an expert in the anti-crime organisation, or by specialist anti-phishing organisations and volunteer groups (Shaikh, Shabut and Hossain, 2016).

The purpose of the present research work is to develop a system to detect phishing attacks and also to provide insights and increase awareness of how active Internet users can protect themselves against such phishing attacks. The outcomes of this study will, therefore, help researchers and industry professionals in identifying trends and in formulating practical preventive measures against cybersecurity attacks. Hence, this study aims to use an algorithm (ANFIS) and website hybrid features to detect phishing activities and protect users while they are surfing the Internet. These features will be used to develop an automated plug-in web browser in an attempt to protect online users against phishing website attacks by classifying the collective properties of the images, frames and text web page features. The real-time activity means that the decision is made by the plug-in to notify the user about the website before the user web browser loads the intended page, a time period which is on average 60 seconds (Barraclough *et al.*, 2013). The developed plug-in checks these features on the web page to determine the level of originality of the website. If the web page check results in a legitimacy level of 85% and above, the web page is considered legitimate according to Barraclough, Sexton and Aslam (2015), whereas at around the 50% level it is deemed suspicious and at 25% and below it is identified as a phishing website. To

the best of knowledge, this is the first work that considers how best to integrate images, text and frame of the website in a hybrid feature-based solution for a phishing detection scheme.

1.3 Motivation and Scope

The use of technology for fraudulent activities has flourished in recent years. The technical resources required to carry out phishing attacks are readily available through private and public sources. Hence, some of these technical resources have been automated and streamlined, thereby allowing their use by non-technical criminals. This automation has made it easier for a larger population of less-sophisticated criminals to commit crimes online, as it has made phishing more viable and economical. According to a report by the Anti-Phishing Working Group (APWG)[1], the number of phishing attacks on website discovered in the second quarter of 2019 was up 36% over the fourth quarter of 2018 (see Chart 1-1), while the most targeted sector is the software-as-a-service (SaaS)/webmail which accounted for 36% of phishing attacks over the same period, followed by payment service sector with 22%, financial institutions with 18% and other sectors with 9% (see Chart 1-2) (APWG, 2019). Phishing attacks pose a severe threat to the economy and security globally (Crosman, Quittner and Wolfe, 2012). Hence there is a need for the automation of phishing detection algorithms, which is within the scope of this research. According to a Financial Fraud Action[2] UK report for 2017, about £165 million was lost to fraud related to Internet payment cards, telephone banking and identity theft in 2017, and while this is 3.68% lower than in the same period in 2016 (Financial Fraud Action, 2017), it is still a cause for concern.

In the recent times, there has been a considerable increase in the assortment, technology and complexity of phishing attacks in response to the increase in

[1] APWG report [online]. Available at: https://docs.apwg.org/reports/apwg_trends_report_q2_2019.pdf [Accessed 31 October 2019]

[2] Financial Fraud Action UK [Online] Available at: < https://www.financialfraudaction.org.uk/wp-content/uploads/2016/07/2017-half-year-fraud-update_September-17.pdf> [Access 15 December 2017]

countermeasures and user awareness in order to sustain profitability from the illegal activities by the phisher (Sharma, Meenakshi and Bhatia, 2017). Providing the ability to detect website phishing attacks may help individual users or organisations in identifying legitimate websites. The effectiveness in recognising an attack may significantly contribute to the making of an effective decision between a fake and legitimate site (Arachchilage, Love and Beznosov, 2016).

Despite various methods having been used to develop anti-phishing tools to combat phishing attacks, these methods suffer low accuracy (Sharma, Meenakshi and Bhatia, 2017). Therefore, there is still room to improve the accuracy of phishing website detection. In the solution proposed in this study, three critical features of image, frame and text of websites are extracted and used for phishing detection. Previous methods have failed to combine the usage of frames, images, and text to develop an effective phishing detection method. Because using only text which is the common trend to a detection phishing website, this will not be effective as some changes can be made to the frame and the image. Doing so is, therefore, the focus of this research work and therein lies its originality using Adaptive Neuro-fuzzy Inference System (ANFIS) as classification algorithm, as well using the deep learning of Convolutional Neural Network (CNN) and Long Shorth-term Memory (LSTM) for further classifying in this solution.

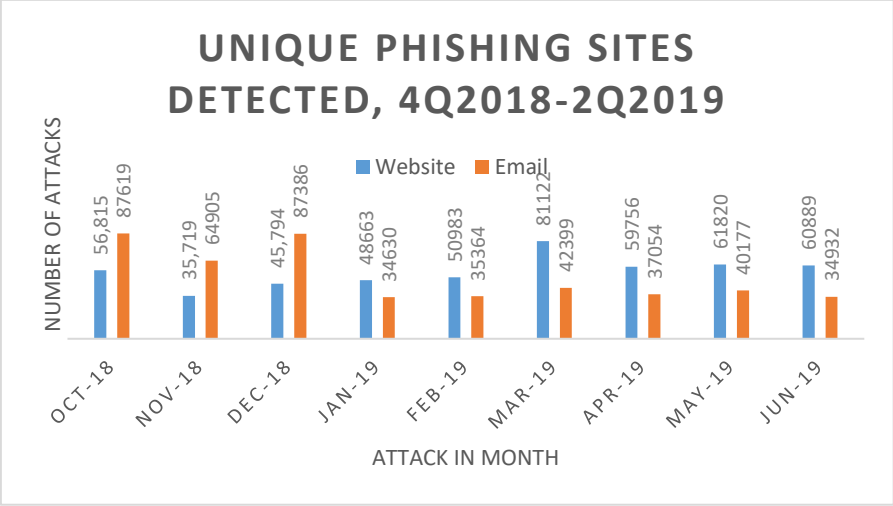


Chart 1-1: Unique Phishing Sites Detection (APWG, 2019)

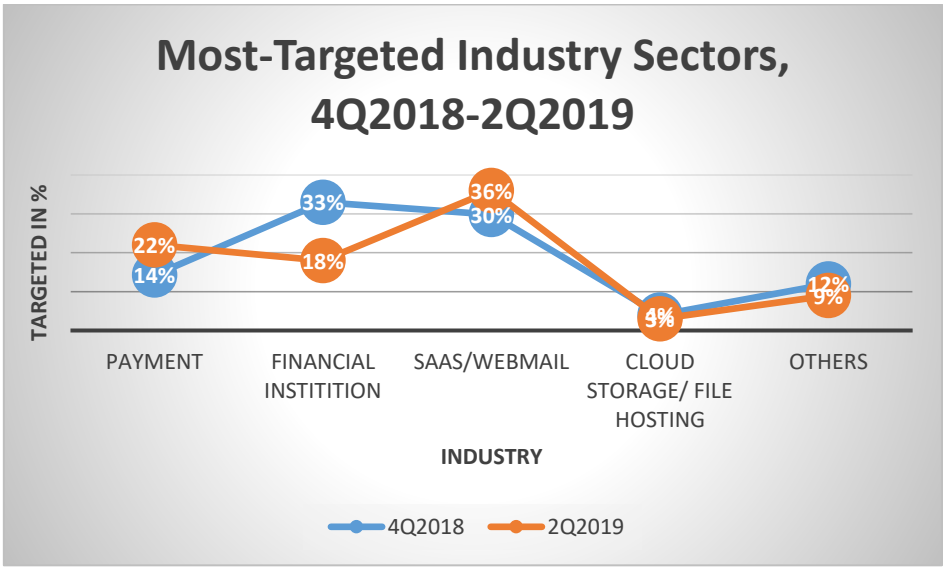


Chart 1-2: Most Targeted Industry Sectors (APWG, 2019)

1.4 Research Questions

This research attempts to solve the phishing website problem by developing a method to detect phishing websites based on the image features, frame features and text features of web pages. The research also looks at how features are extracted from a website and how they can be used to automate the detection of a phishing attack using a browser extension. In this thesis, we attempt to answer the following two research questions:

1. *How can we detect a phishing website effectively?* Detection plays a significant role in tackling phishing attacks. In this research, an algorithm with website hybrid feature will be use to developed a decision support plug-in to effectively detect a phishing website attacks.
2. *How can we automate phishing detection?* Automate the plug-in in MATLAB version 9.5 AppDesigner and test the application on dataset to evaluate the effectiveness of the plug-in.

1.5 Research Aims and Objectives

The main aim of this research is to develop an intelligent phishing detection and protection scheme for identification of website-based phishing attacks. This goal involves improving on previous work by building a robust classifier for intelligent phishing detection in online transactions. In order to achieve this aim the intelligent phishing detection support system should possess the following characteristics:

1. **Robustness**: It should have a hybrid algorithm that can support efficient classification for website phishing detection in real-time.
2. **Accuracy**: It should improve accuracy by reducing the false positive (FP) rate and increasing the true positive (TP) rate with absolute precision.
3. **Optimisation:** It should be able to optimise performance by employing a hybrid method that uses the features of website images, frames, and text for the user's objectives.
4. **Real-time functionality:** It should notify the user about the about the legitimacy of the website before the user web browser loads the intended page.

These requirements will be met by achieving the following five specific objectives:

I. Examine the Adaptive Neuro-fuzzy Inference System (ANFIS) algorithm as a baseline and the use of more advanced methods to improve accuracy.

II. Develop an algorithm that improves phishing-detection accuracy by comparing the text, images and frames of a given website with a knowledge model;

III. Train, test, and validate the developed system (machine learning) for real-time phishing detection;

IV. Automate the detection mechanism in real-time and test it offline.

V. Develop a plug-in and implement on a cross-platform operating system;

1.6 Contributions to Knowledge

Phishing website detection has become the focus many studies and has been considered by many researchers. However, phishing has become more complex, and attackers can now bypass the filters that have been put in place by anti-phishing systems (Baykara and Gürel, 2018). Phishers sometimes implement new techniques such as embedding obfuscation in a website URL, creating a hyperlink from the original website or redirecting their victim to a phishing web page by using malicious software. Hence, it is necessary to identify phishing behaviour in online activity to detect phishing websites. In this regard, this research makes the following contributions:

1. First, a hybrid feature selection approach is developed for use in the detection of website phishing attacks. The proposed method is based on a combination of content-based and visual-based approaches. The hybrid feature selection approach combines the use of the associated elements of images, frames and text of legitimate and non-legitimate websites and an associated artificial intelligence algorithm to develop an integrated method to address these elements together. The current methods to classify phishing websites are based on modest features of phishing attacks such as text, but these are not sufficient to combat the threat (Sharma, Meenakshi and Bhatia, 2017). The proposed phishing detection and protection scheme is based on the ANFIS algorithm, which is a robust scheme and uses integrated features of frames, text and images for web-phishing detection and protection. The deep learning (DL) algorithm is evaluated as a baseline to build the solution. The use of three features of a website (image, text and frame) make it more efficient in phishing detection, rather than using only one element for detection as a single solution.
2. A second notable contribution of this study is that it is the first study to attempt to differentiate legitimate from phishing websites using two deep learning techniques, the Convolution Neural Network (CNN) and the Long short-term

memory (LSTM) approach, that are combined as a single classifier in a novel approach called the Intelligent Phishing Detection System (IPDS).

3. Thirdly, a total 13,000 features and 10,000 images were collected both from phishing and legitimate websites providing an important new dataset resource for other researchers in phishing detection to utilise (Adebowale, 2019).

1.7 Thesis Structure

The chapters of this thesis are structured as follows:

1. Chapter 1: Introduction. This chapter introduces the issue of interest and the significance of this research study. It provides details of the research problem and the research questions to be resolved together with the precise research objectives. It also summarises the existing literature and clarifies the main contributions of this research.
2. Chapter 2: Literature Review. This chapter contains a review of the literature on the topic under study, namely phishing detection schemes. It also discusses the focus of the research by critiquing the relevant existing research methods and summarising their findings as well as their strengths and weaknesses. It then discusses appropriate provision for the phishing detection problems and how to resolve them.
3. Chapter 3: Research Design and Methodology. This chapter provides details of the methodological approach used in the research and justifies the selection of the chosen methods. Also, it describes the features collection, the size data and sources from which the features are collected. It also describes the optimisation concept of the methodology as well as the feature preparation and normalisation steps. This chapter also provides detail about deep learning algorithm, the hybrid approach to detect phishing website using Long Short-Term Memory (LSTM) and Convolution Neural Network (CNN) with the structure of both algorithms.

4. Chapter 4: Implementation of Intelligent Phishing Detection System (IPDS). This chapter presents the experimental work and process done in the research using the adaptive neuro-fuzzy inference system (ANFIS) methodology with three sets of input as a hybrid. Besides, it details of the process used for fuzzy modelling and fuzzy operation in the testing, training and validation phases. It then presents the results of applying the proposed offline approach. Also, it presents the experimental work done in the research using the CNN+LSTM model and how the system work. It then justifies the method and explains its limitations. The chapter also presents the approach used in developing our model with the result of the experiment.
5. Chapter 5: Implementation and Evaluation of Phishing Detection Toolbar in MATLAB version 9.5 AppDesigner. This chapter proposes an approach for a web browser toolbar for phishing detection. It discusses the result of the validation test of the toolbar using a suspicious site, legitimate website and phishing websites from various sources. It also provides a comparative study to highlight the benefits and limitations of the proposed plugin.
6. Chapter 6: Conclusion and Future Work. This concluding chapter contains a summary of research achievement and the contributions of this study to knowledge, with some future research directions.

CHAPTER 2: LITERATURE REVIEW

Chapter 2 Literature Review

2.1 Introduction

The previous chapter introduced the research topic and outlined the subject area to be covered in this thesis. This chapter discusses the background and basic concept of phishing website detection with a focus on the feature-based approach (of particular interest to this present study).

2.2 Background to the Phishing Problem

The principal approach that phishers adopt to conduct their nefarious activities involves presenting fake situations to potential victims in which the users are directed to take a certain kind of decisive action. For example, they may send an email stating that the user's bank account requires an urgent update due to some security measures, or stating that a user's transaction has not been processed due to incorrect information that the user presented while purchasing their goods online (Bandhaniya and Joshi, 2017). The anti-phishing filtering system has been reinforced to withstand massive generic phishing attempts, and user awareness has increased regarding the existence of generic phishing that targets both organisations and individuals (Gascon *et al*., 2018). Nevertheless, the attacker's ability to forge customised messages to send to their targets still results in a high success rate as these messages can exploit human vulnerabilities (Aleroud and Zhou, 2017). One of the more sophisticated phishing attacks is the spear-phishing attack (Bender *et al*., 2018). In this type of attack, the attacker creates fake content with relevant information based on the personal data of the target, which make the information look more legitimate to the user when they are visiting the phishing website (Steer, 2017). This type of attack is not readily identifiable by non-technical users, and even those well versed in technology have difficulty in spotting this type of fraudulent scam (Deshmukh and Popat, 2017).

Phishing is a term that is commonly used to define a scam that uses spoofed web pages and unsolicited emails, which are sent by a phisher to lure victims into revealing personal information (Zhao *et al.*, 2017). The term phisher is used to describe the criminal individual who uses the information acquired through phishing to, for example, steal money from a victim's bank account. They might also use the information to access the victim's computer, lock them out, and then demand a ransom from their victim in return for handing back control (Pathak and Nanded, 2016).

Phishers use various techniques to gain personal information from users. For instance;

- The phisher pretends to be the victim's bank, sending them an email message informing them that their bank account details have expired and that there is a need to update them. The potential victim is then instructed to click a link in the body of the email so that the victim can continue to use their account (Crain, Opyrchal and Prakash, 2010).
- The phisher pretends to be the victim's bank and sends them a message stating that there has been a suspicious purchase made with their bank card. They suggest that the victim clicks on the link in the email if they would like to cancel the transaction.
- An email message is sent to the victim claiming that they have won a lottery, and states that in order to claim the prize, the victim needs to click on the secure link provided, enter their bank account details and the money will then be transferred into their account.
- The Phisher pretends to be the inland revenue, informing the victim by email that they are due a tax refund due to an overpayment of income tax, and must provide their bank details in order to process the refund (Khadir, 2015).

Phishing attacks are divided into two groups: (1) ‘flash attacks’ that give the attacker a virtually undetectable way to redirect the victim to a fake website (an example of this is the so-called “man in the middle” attack) and (2) ‘non-flash attacks’ in which an attacker uses existing software to enable applications and authorised protocols to carry out malicious activities, which can only be revoked by making a payment. Such an attack is called ‘ransomware’. Flash attacks usually involve the dissemination of a large volume of similar messages that are transmitted within a short period of time (Bullee *et al.*, 2017), whilst non-flash attacks although similar deliver messages over a relatively long period of time (Lin *et al.*, 2015). The interaction between the receiver and the message may happen when the receiver follows the malicious link in the message they have received, replies with useful information or fills in a deceptive form with relevant data, which then allows the attack to succeed. Regardless of the type of attack it follows the same process flow ((Ahmed and Abdullah, 2016) as shown in Fig. 2-1 and which consists of five phases as follows:

1. **Planning**: First, the phisher will identify the target user or organisation that they wish to be their intended victim. Next, they will decide how to get the personal information of their targets, such as their account number, password and email address. Then a phishing website is structured as a clone of the original website so that the victim is not able to distinguish it from that of the service they usually access (Virvilis *et al*., 2014).
2. **Setup**: After the phisher has decided on their target, they set up and prepare to attack. In this phase, the phisher creates techniques for sending many spoof email messages to random Internet users that seem to be coming from a legitimate and well-known organisation (Kazemian and Ahmed, 2015). The email urges the victim to update their personal information to avoid losing access rights to specific services, and the phisher thereby collects valuable information from their target. Usually, they do this by developing a web page and email addresses.

3. **Attack:** The attack phase involves delivering a malicious payload through three common propagation vectors, either by a deceptive message, spam email or fake website (Silic and Back, 2016). Ordinarily, the phishing message appears to be emanating from a reliable source. Hence, the victim may take action that compromises their personal information; by clicking on the link provided the victim is directed to a bogus website implemented by the attacker, or the user is prompted to provide confidential information, either by a delivered web Trojan on their system or a remote web page (Amrutkar, Kim and Traynor, 2017). Spam filtering can block many of the phishing emails; if an organisation whose user is being phished uses authenticated email regularly, the email recipient may notice that the email does not have a valid signature, which thereby stopping the attack (Gascon *et al.*, 2018).
4. **Collection**: The personal information that the user inputs into the fake site is relayed to the phishing system and the phisher. This compromised information is obtained through a pop window or web pages presented to the user.
5. **Identity theft and fraud**: The final phase of the phishing attack involves the phisher using the sensitive data gathered from the victims. This information is used to impersonate the victim and making illegal purchases or to commit other types of fraud. Phishers evaluate their attack, and if it has been successful, they repeat the same process over and over again.

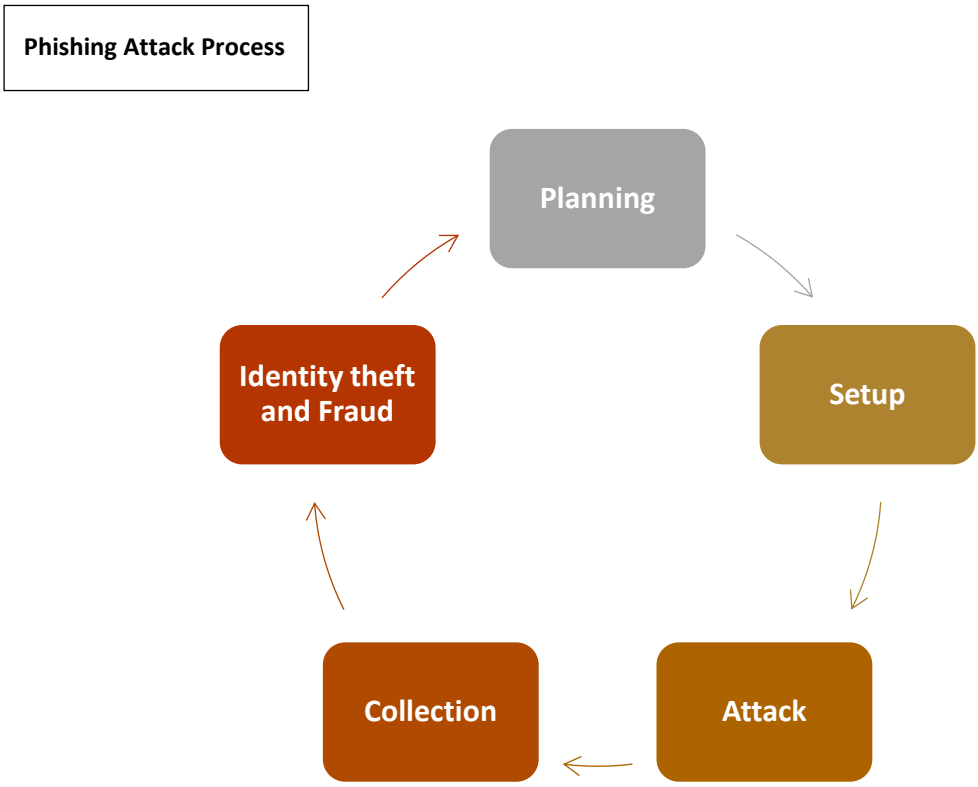


Fig. 2-1: Phishing Attack Process

The use of Hypertext Transfer Protocol Secure (https) as a website prefix indicates the communication is secure as it denotes that the data exchanged between a person and the website they visit is encrypted and is a considerable measure to protect malicious site interactions. The https protocol is always used by websites that offer online sales or to protect user passwords. However, a study by the Anti-Phishing Working Group (APWG 2019) provides insights on how phishers are fooling Internet users by turning Internet security features against them. The report states that in the second quarter of 2019, more than a third of phishing attacks were hosted on websites that had a Secure Sockets Layer (SSL) certificate and https. Moreover, a phishing page can still function as intended, even without an SSL certificate (APWG, 2019). Phishers take a further step to make their site look legitimate by obtaining a valid SSL certificate as well creating an https page to make their victims think that the site is the original and this may, therefore, lead to a successful attack. The general widespread misunderstanding of the meaning of the https designation and the confusing labelling

of https websites on web browsers are the primary reasons why the usage became a popular favourite for phishers to host phishing sites (Thakur and Kaur, 2016).

More sophisticated types of phishing attack have been identified in recent times and some of the main ones are listed below:

- **Session hijacking**: This type of attack occurs when a user's activities are monitored online until they log into their target account or transaction and establish their official identifications (Deshmukh, Popat and Student, 2017). At this point, the malicious software takes over and can perform illegal actions such as bank transactions without the user's knowledge.
- **Screen loggers and key loggers**: This type of attack usually involves the use of a variety of malware that tracks keyboard input and sends relevant information to the phisher via the Internet. The phisher can embed malware into the user's browser as a small utility program, known as helper objects, which runs automatically when the web browser is started, as well as into system files as a screen monitor or device driver (Gascon *et al*., 2018).
- **System reconfiguration:** This attack type usually modifies the user system settings for criminal purposes. The inclusion of a fake URL in the user's favourites file may be modified to direct the user to a web page that looks like the legitimate site.
- **Data theft:** Unsecured access server can be accessed with the techniques to steal sensitive data from an organisation. This type of attack is also called espionage and encompasses stealing confidential information, research and development, legal opinion and employee-related records. The phisher can profit from obtaining these types of sensitive information as its disclosure can cause embarrassment or economic damage to the victims (Oest *et al.*, 2018).
- **Pharming**: This type of attack is based on a Domain Name Server (DNS) phishing attack or host file modification. A pharming attack occurs when a phisher tampers with an organisation's host file or the DNS so that users

request to the website name service returns a compromised address, and subsequent communication is directed to the fake website (Purkait, 2015). Although the users will be unaware that the fake site is where their confidential information is submitting and the phisher controls over it.

- **Content Injection**: This type of attack involves replacing some part of the content of the legitimate website with fake content to mislead the user into giving their confidential information to the phisher (Silic and Back, 2016). The phisher inserts malicious code into the real site to log the user's information and credentials which are then secretly delivered to the phisher's server.
- **Man-in-the-Middle**: This attack is one of the more sophisticated attacks and is more difficult to detect than many of the other phishing attacks. In this type of attack, the phisher positions their fake site in between the user and the legitimate site. They collect the information that is entered into the fake site but then pass it on to the legitimate site, so the user's transaction is not affected (Ahmed and Abdullah, 2016). In this way, the phisher can gather the information in order to sell to fraudsters that use them when the user is not active online.

2.3 Overview of Anti-Phishing Techniques

Phishing attacks usually involve the use of complicated tricks, which makes it problematic for users to know if they are the victim of phishing (Abbasi *et al*., 2015). Legacy anti-phishing techniques can be categorised into (1) threat elimination and (2) user awareness and education, the latter of which is aimed at educating users so that they do not become victims of phishing attacks (Gavahane *et al*., 2015). Existing phishing detection tools have improved on legacy tools, but they still suffer from false negatives, that is to say, false alarms. Also, regrettably, malware scanners are not very good at spotting malicious objects. Most virus scanners apply a fingerprint technique to detect phishing activities (Safer-Networking, 2016). This technique involves

scanning the binary code of the known malicious pattern. However, this detection strategy has a poor response time against phishing attacks that utilise malware (Safer-Networking, 2016).

The use of a personal firewall gives an added layer of security against phishing by imposing control over network traffic; the firewall is designed to defend the network against malware-based phishing attacks, such key loggers and Trojans. Such attacks always have to transmit the captured information over the network through the Internet to the phisher. These phishing activities can be prevented by using a personal firewall (Thiyagarajan, Venkatesan and Aghila, 2010). Moreover, authentication mechanisms play a significant role in combatting phishing attacks. A user authentication mechanism helps in validating the identity of the user and their location, and the server also needs to be authenticated (Silic and Back, 2016). Nevertheless, phishing attacks still work because they exploit the user's social instincts, such as being helpful and efficient. Hence, these attacks can be particularly powerful because these instincts also make us good at our jobs and should not be discouraged. Thus, educating users to identify suspicious websites and emails is essential in defending against phishing attacks as is providing documentation to increase their awareness (Jansson and von Solms, 2013). However, users are not usually interested in reading documents about anti-phishing, especially when surfing websites as security is not their primary concern. The majority of users do not find anti-phishing documents, new reading material. On the other hand, it has been shown that the use of game-based learning increases user motivation and awareness (Yang *et al.*, 2012). Therefore, various anti-phishing games have been developed to educate users about this threat (Arachchilage and Love, 2013; Yang *et al.*, 2012; Arachchilage, Love and Beznosov, 2016). However, the effective prevention of phishing attacks requires a solution that combines process, technological and people-based approaches (Comar *et al.*, 2013). These approaches must be considered together to create a holistic defence strategy to protect against phishing attacks and also to encourage users to report suspicious websites and emails, but

there is need to back that up with a technical support in doing that, and timely feedback is submitted back to the user (National Cyber Security Centre, 2018).

2.3.1 Anti-Phishing Techniques

Whilst phishing can be channelled via social media, text message or by phone (Babu, Nirmala and Kumar, 2010) nowadays most people use the term phishing to define attacks that arrive by email. Email is a perfect distribution technique for a phishing attack as it can be disseminated to numerous users straightaway and hide among the vast number of useful emails that busy users receive (Bandhaniya and Joshi, 2017). Phishing emails can hit an organisation regardless of its size or type. Apart from the theft of information, attacks can install malware, disrupt organisational systems, or steal money through fraud. An organisation might get caught up in a mass operation where the phisher is just looking to gather some new passwords or make some easy money, or the aim could be something much more precise such as the theft of sensitive data. Using the spear-phishing attack (Fig. 2-2) against an organisation is when a phisher uses information about an organisation to make their message more persuasive and realistic, as reported by the (Government Communication Headquarter. (GCHQ, 2018).

The primary defences against phishing are reliant on users' abilities to detect phishing emails. By strengthening and broadening the available defences, an organisation can improve its resilience against phishing without disrupting the productivity of users (Government Communication Headquarter. (GCHQ, 2018). However, accepting the fact that some phishing emails will get through will help organisations to plan for the day when an attack is successful, and also minimise the damage caused (Bandhaniya and Joshi, 2017). Below are some steps that can be taken by organisations to build effective defences against phishing attacks:

1. Make it difficult for attackers to reach users

2. Help users identify and report suspected phishing website/emails
3. Protect the organisation from the effects of an undetected phishing site
4. Respond quickly to incidents.

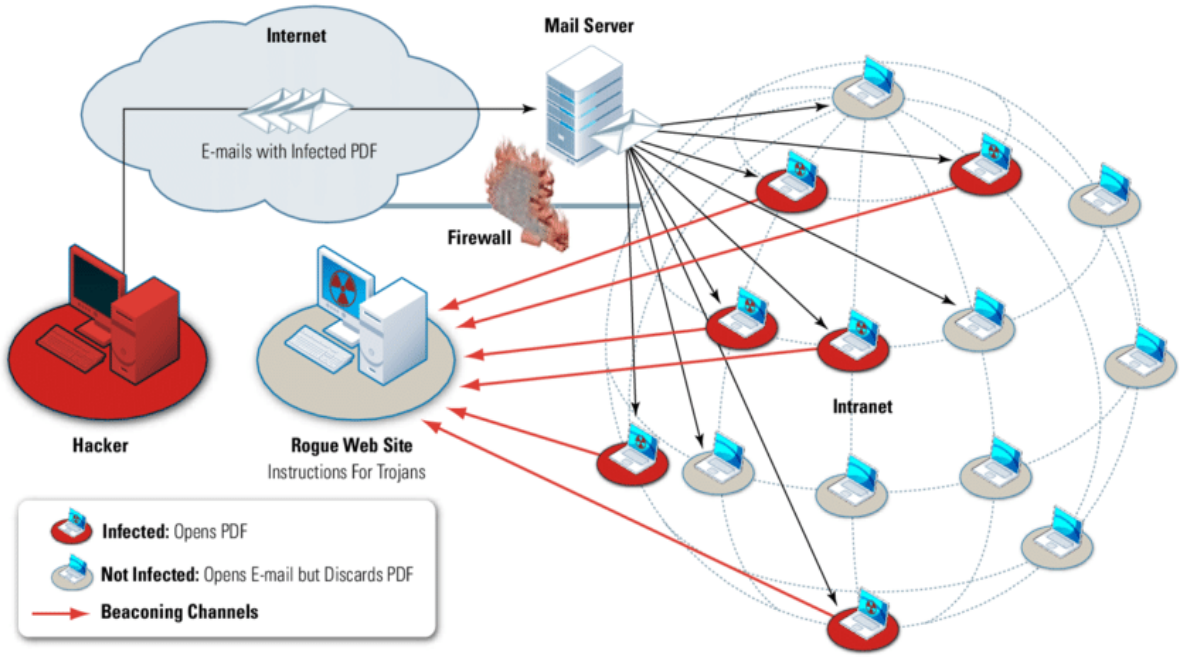


Fig. 2-2: Spear-Phishing Attack (Source: Karen Goertzel, 2012)

Several methods have been proposed in recent years to overcome phishing attacks (Al-Daeef, Basir and Saudi, 2014). Khadir and Sony (2015) reviewed a range of phishing detection techniques that had been proposed up to the year 2015, mainly focusing on scientific techniques and machine learning technology. They categorised these techniques into technical and non-technical approaches. The technical approach includes heuristic-based, content-based, blacklist/whitelist-based and toolbar approaches.

2.4 Technical Phishing Detection Solutions

Phishing detection strategies are implemented in a reactive and proactive manner against phishing attacks (Oest *et al.*, 2018). This detection strategy has been

implemented in various anti-phishing tools. As most anti-phishing solutions run many checks, they can detect more phished websites than others. However, when multiple checks are performed to validate a web page, this can result in a slow response time, which could make the users of that solution frustrated and reduce their usage. In order to prevent an attack on the user doing their legitimate activities online, there is need to completely adopt appropriate security measures. One approach is to set up a dummy e-commerce website account and monitor and/or protect the account from phishing activities. Many solutions for stopping phishing attacks rely on checking the content and the visual similarity in order to expose phishing sites that obscure their online identity.

2.4.1 Content-Based Approaches

Thiyagarajan, Venkatesan and Aghila (2010) proposed a method that addresses one of the limitations in the transaction authentication number (TAN) method. The proposed method was based on the TAN approach but included a modification to the challenge and response techniques. However, their method protects against phishing attacks in nearly the same manner as the TAN method except for some intelligence added for providing the challenge-response (Thiyagarajan, Venkatesan and Aghila, 2010). Also, Blum *et al*. (2010) proposed a solution that utilises a confidence-weighted classification combined with content-based phishing URL detection of present and emerging types of phishing domains (Blum *et al*., 2010). The proposed approaches are better than comparable approaches. However using only the content of a website only will not be surfficient to protect user against phising attack, but it requires additional features such as image and frame to make it more effective.

The solution proposed by Dunlop *et al*. (2010) consists of three main steps and involves capturing the image of the website of interest in a user's web browser as an image and using OCR techniques to convert the images into readable text. They tested their solution on 100 websites and found that it was predicted 98% true positives and

2% false positives. Their solution is better than the previous image comparison methods that rely on a database. Nevertheless, more comprehensive tests need to be carried out because 100 sites are minimal in comparison to the number of phishing sites that emerge every day (Sharma, Meenakshi and Bhatia, 2017). A scheme was also proposed by Dunlop, Groat and Shelly (2010) called GoldPhish, which provides zero-day protection phishing attacks with high detection rate (Dunlop, Groat and Shelly, 2010). However, their method has unstable performance, using a third party database, if an image is manipulated with little variation; this will causes the detection process to fail inaccuracy.

Also, Alkhozae and Batarfi (2011) proposed a solution based on checking the web page source code in which the security of the website is evaluated by checking the characters in the web page source code for phishing characters (Alkhozae and Batarfi, 2011).

A scheme was also proposed by Dunlop, Groat and Shelly (2010) called GoldPhish, which provides zero-day protection phishing attacks with high accuracy (Dunlop, Groat and Shelly, 2010). However, their method has unstable performance if such an image is manipulated with little variation this causes the process to fail in accuracy. Also, Wardman *et al*. (2011) proposed a solution that depends on a file-matching algorithm that is implemented to detect a phishing website based on its content. To test their solution, they used a dataset of 17,992 phishing attacks targeting 159 different brands. The tests on the file-matching and string-alignment techniques were done on MD5 matching, deep MD5 matching, and syntactical fingerprinting. Based on the results of their experiment, the authors report that syntactical fingerprinting outperformed file-matching in regard to the detection rate. Moreover, the full implementation using syntactical fingerprinting for candidate file selection had the best overall error rate handling performance. The authors report that the result of the experiment using a variety of different content-based approaches demonstrated that some were able to achieve a 90% detection rate (Wardman *et al.,* 2011).

Likewise, Aggarwal, Rajadesingan and Kumaraguru (2012) developed a Chrome browser extension that detects phishing on Twitter in real time. They use the content of the tweet and some URL features such as hashtag, length, and mentions, and apply machine learning classification to detect phishing. The result is promising, but the solution needs to be tested on other social media to produce a more robust result (Aggarwal, Rajadesingan and Kumaraguru, 2012).

On the other hand, Mao *et al*. (2013) proposed an algorithm to quantify the suspicious ratings of web page layouts by using CSS as the basis for detecting visual similarities among web pages. The prototype extension, called BaitAlarm, was used against 7,000 phishing websites. The authors report that the scheme was able to achieve 96% detection rate in thousands of the samples used for their experiment (Mao *et al.*, 2013). Chang *et al*. (2013) proposed a method that extracts a screenshot of the web page and segments the region of interest, which includes the website logo for phishing detection. However, their solution may be more effective if they use more features to make robust as the scheme only relies on the image of a web page. In a different vein, Fatt, Leng and Nah (2014) proposed an approach that employs the website favicon to search for the identity of a website and uses the Google search-by-image API search engine solution in order to evaluate the authenticity of a website. The authors used 1,000 web pages to verify the effectiveness of their approach. The results showed that it achieved a better of 97.2% true positives and 5.4% false positives. Their approach is efficient because it does not maintain an image database for its operation (Fatt, Leng and Nah, 2014). However, it could achieve even better results if it were combined with the use of other phishing detection methods that consider features such as the text and the frame structure according to the author. Their approach does have capacity to include more features, which would make it a robust solution as reported in their study.

Kumar and Kumar (2015) develop an anti-phishing solution based on a visual cryptography approach. In their method, the user generates two shares of images

using a (2, 2) visual cryptography scheme. The first time that the user registers on a website, the user stores the first share of the image, and the other part is uploaded to the site. During each login attempt, the user must verify the legitimacy of the location by comparing the image of both shares (Kumar and Kumar, 2015). However, their test result is not robust due to the low number of websites used in the test experiment, so there is a need to undertake a comprehensive analysis to improve precision.

Shekokar *et al*. (2015) proposed a solution for the detection and prevention of phishing that involves web page similarity and URL-based detection. They used the LinkGuard[3] algorithm to analyse the extracted URL from which the website is directed and the virtual URL that is seen by the user (Shekokar *et al*., 2015). Unfortunately, their result is not reliable due to the hundreds number of websites used in their experiment, so there is a need to do a comprehensive analysis using thousands of websites to improve precision.

Also, Kazemian and Ahmed (2015) proposed a novel approach to detect fake web pages that are based on the utilisation of machine learning algorithms. Their experiment showed that the supervised learning techniques were able to produce up to 98% classification accuracy (Kazemian and Ahmed, 2015). However, their online approach was unable to use an incremental data source. They employed a different approach than traditional batch processing to accommodate new incoming data by using stream data in the form of a list of phishing web pages and safe websites. The approach is better than existing comparable ones,because of it zero day protection approach. However, relying only on one feature of a website is not safe as phisher can use a deciful means to alter the image of an original website, and use it to create phishing web page. But Including an additional feature and multiple processing will make it more strong in detection.

[3] LinkGuard algorithm [Online] Available at: http://www.ijafrc.org/Volume3/issue34/6.pdf [Accessed 20 October 2016].

Gavahane *et al*. (2015) proposed a system that focuses on retrieving the necessary attributes in real-time using Hadoop-MapReduce, which helps to increase both the speed and throughput of their model. The goal of their system is to increase the speed of phishing detection (Gavahane *et al.,* 2015). On the other hand, Chiew *et al.* (2015) proposed a method for detecting a phishing attack in which a website logo image is used to determine the identity consistency between the legitimate and the phishing website. The experiments gave reliable and promising results (Chiew *et al*., 2015).

Tan *et al.* (2016) proposed a phishing detection technique based on the identification of the difference between the target and the original web page. The proposed method is called PhishWHO and consists of three phases. Their overall experiment results showed that the scheme outperforms most of the conventional phishing detection techniques with which the proposed scheme was compared (Tan *et al*., 2016). In a similar vein, Hu *et al*. (2016) proposed a new phishing detection method based on the analysis of a legitimate server's log information. Their idea is based on finding the references point that is used every time the victim opens the phishing website; the site will refer to the original website by asking for resources. The authors report that the result of their experiments showed that their proposed scheme is both highly accurate and effective (Hu *et al.,* 2016).

Marchal *et al*. (2016) developed a phishing detection scheme that contains several essential elements and has several advantages: it requires very little training data, scales well to much larger test data, is language independent, is resilient to adaptive attacks, is fast and can be implemented as a client-side solution (Marchal *et al*., 2016). Li, Yang and Ding (2016) proposed a novel approach that uses the minimum Enclosing Ball Support Vector Machine (MEB-SVM) to detect a phishing website. They aimed to improve the speed and accuracy with which phishing websites are detected (Li, Yang and Ding, 2016). The proposed model was able to predict the presence of a phishing web page with 96.6% accuracy, which is 0.4% higher than the SVM classifier with 96.2%. The performance of the model, therefore, improves on that of the SVM

algorithm. However, in their model, only a small amount of data was used to build the classifier, and it only consisted of text features. Therefore, this model can be improved by using a larger dataset and other web page features such as frames and images to increase accuracy further.

Zouina and Outtaj (2017) proposed a phishing detection solution based on using six features of the URL and used an SVM and similarity index to perform the detection procedure. The result showed that the method was able to predict the presence of a phishing URL with 95.8% accuracy. The authors stated that detection could be improved further by using a larger dataset and more features because the number of features they selected is small compared to those that the phisher would use in creating a fake website (Zouina and Outtaj, 2017). Marchal *et al*. (2017) presented an approach for detecting phishing web pages in real-time while they are visited by the browser (Marchal *et al*., 2017). The add-on uses JavaScript and interacts with the web browser. The authors tested their solution to 8,500 legitimate and 1,500 phishing datasets, and it achieved 98% precision. This phishing detection algorithm could also be improved by using more techniques because the number of features it considers is small compared with those that the phisher can employ by using various tools to create look-alike websites.

Mao *et al*. (2017) proposed a solution called the phishing-alarm to detect phishing attacks by using features that the phisher has to use to create a fake website that looks like the original one. They also proposed an algorithm to quantify the suspicious ratings of a web page that works by identifying the visual similarities between web pages. Their solution was prototyped and implemented in the Google Chrome browser (Mao *et al*., 2017). Jain and Gupta (2017) proposed an approach that utilises a feature set containing features such as HTML tags, text formats, text contents, CSSs and images to decide as to whether a website is suspicious or not (Jain and Gupta, 2017). Zhang *et al*. (2017) introduced prediction label of website contents to be part of the proposed framework for phishing web page detection using hybrid features that include textual

content, URL and rule-based features. Their framework was developed with the use of an efficient two-stage extreme learning machine (ELM). Their solution is highly accurate, but it could still be improved by adding other features to make it more robust (Zhang *et al.*, 2017).

Churi *et al*. (2017) proposed a model that uses visual cryptography and code generation techniques as methods of authentication. Their scheme ensures that users know that the sites they visit are legitimate, especially those with which they are already registered, and makes users aware of phishing attacks (Churi *et al*., 2017). On the other hand, Tripathi, Nigam and Edla (2017) proposed a new architecture for web fraud detection using the apriori algorithm for association rule mining with the PhishTank dataset in the web advertising network. They analysed the web access log which stores the activities performed by end-users. This log is used to detect a fraud sequence of repeated web URLs (Tripathi, Nigam and Edla, 2017). Patil, Rane and Bhalekar (2017) also develop a solution that uses the basic visual features of a web page's appearance as the basis for detecting page similarities. They implemented the obfuscated URL detection algorithm, which provides multilayer security that protects user data against phishing attacks over the Internet (Patil, Rane and Bhalekar, 2017).

Mishra and Gupta (2018) proposed a novel intelligent phishing detection system to detect zero-day phishing attacks using the concept of uniform resources identifier (URI) and cascading style sheet (CSS) matching. The authors report that the proposed solution is very efficient in detecting phishing and zero-day attacks with a true positive rate of 93.27% (Mishra and Gupta, 2018). Likewise, Smadi, Aslam and Zhang (2018) proposed a novel framework that combines a neural network with reinforcement learning to detect phishing attacks in real-time online for the first time. The authors report that their proposed model can handle zero-day phishing attacks with high performance achieving a true positive rate of 98.63% (Smadi, Aslam and Zhang, 2018). The above approaches are useful but use text features only in their prediction,

which can be improved if the other aspect of the website features such the frame and image is added for more accurate detection.

Shirazi, Bezawada and Ray (2018) proposed a solution that indicates the domain name of phishing websites reflect of phishing site and holds the key to successful phishing detection. The authors report that their model was able to achieve 97% classification accuracy (Shirazi, Bezawada and Ray, 2018). Kuo, Lee and Lee (2018) also proposed a method for identifying phishing websites in which the content of a website that the user visits are extracted and translated into a format that can be classified (Kuo, Lee and Lee, 2018). On the other hand, due to the increase in the number of phishing attacks in recent times, Butler and Butler (2018) anticipated a solution to educate Internet users to try to stop them becoming the victim of a phishing attack and suggested a framework of anti-phishing measures. Sahoo (2018) proposed an architecture to differentiate between fake and legitimate emails with high accuracy. The author used data mining algorithms to analyse emails and help in preventing phishing attacks (Sahoo, 2018). All the methods rely only on the text content of the website. However, these methods can be improved with the integration of other website features such as image and frame.

Rao, Vaishnavi and Pais, (2019) propose a solution called CatchPhish. The proposed solution is a lightweight application that predicts the legitimate URL without visiting the website. The scheme uses full URL, hostname, TF-IDF features and phishing hinted words from the suspicious URL for the classification using the random forest classifier algorithm. They investigate the URL with the use of hand-crafted features and TF-IDF features. The hand-crafted features include special characters and word in the URL. Also, the TF-IDF features were extracted using the information retrieval algorithm is applied on set URLs. They combine the set of hand-crafted and TF-IDF feature are loaded into the machine learning for classifying the URL. The result of the experiment reflect that their scheme was able to classify the URL with an

accuracy of 94.26% on their dataset and 98.25% on a benchmark dataset (Rao, Vaishnavi and Pais, 2019).

2.4.2 Heuristic Approaches

Heuristic-based anti-phishing techniques use website features for phishing detection analysis to create a robust classification model (Lee and Park, 2016). Some original resilient and efficient models that use heuristics for detecting phishing websites have been proposed (Barraclough *et al*., 2013; Aburrous, 2010; Weiwei *et al.,* 2012). The results of the above-cited works show that there is a considerable improvement in the accuracy of phishing detection using neuro-fuzzy. However, the solutions proposed in those works are text-based, so they could be expanded with other website features to enhance detection.

Meanwhile, Abbasi *et al.* (2010) proposed the development of a new class of fraudulent website detection system that is based on statistical learning theory (SLT), also known as Vapnik-Chervonenkis theory, which is a computational learning theory that attempts to explain the learning process from a statistical point of view (Abbasi *et al.*, 2010). Similarly, Wenyin *et al*. (2010) proposed an approach for finding phishing web pages that depend on the construction and reasoning of the semantic link network (SLN) of the suspicious web page. The authors report that their experiment showed that the method was able to achieve a false negative rate of 16.6% for phishing sites and a high level of accuracy for legitimate sites (Wenyin *et al*., 2010). Aburrous et al. (2010) proposed a model based on fuzzy logic combined with a data mining algorithm to characterise e-banking phishing websites. The model was able to identify worse e-banking phishing site of 83.7% accuracy and best e-banking phishing website of 16.4% accuracy, which representing the legitimate Internet banking site. However, the feature set that was used in their work was based on text-only features, so it needs to be more comprehensive to detect e-banking phishing websites. Although, the model approach

is efficient, the level of accurracy is low deu to the number of dataset used, and some data are not necessary for improving the phishing detection accuracy.

Afroz and Greenstadt (2011) proposed a phishing detection approach that uses the profiles of legitimate websites appearance to build a fuzzy hashing system for phishing detection. They evaluated their approach on over 600 phishing sites that duplicated 20 original sites and showed that it provides the same precision as that offered by blacklisting methods, with the added benefit that it can classify new attacks and targeted attacks against smaller sites. The scheme is likely to have a beneficial impact on phishing detection by reducing the efficiency of the sites that look a lot like real websites, thus giving users a better chance of detecting 'phishy' sites. The authors showed that their model was able to detect a phishing attack with 97% accuracy (Afroz and Greenstadt, 2011). The level of accuracy in their scheme motivate this study to do more in term of phishing detection accuracy.

Huang, Qian and Wang (2012) proposed an anti-phishing solution based on a semi-fragile watermark to provide protection for the online activities of service providers. The semi-fragile watermark consists of website identity characters, the URL, and the heuristics that occur in a phishing attack, which are embedded into the service provider's website tags. When a suspicious web page is launched, the provider uses the solution to compare the embedded characters with the generated watermark, and if an inconsistency is found an alert raised. The identified spoofed website is then classified as a phishing website (Huang, Qian and Wang, 2012). Their solution, which is based on embedded watermark in a web page, could improve the detection rate, but there is a need for more features to make the algorithm more robust and enable comprehensive detection. Their approach was carefully look into and inspire the decision to use frame as part of our hybrid solution.

Barraclough *et al.* (2013) proposed a neuro-fuzzy system with fuzzy rules to differentiate between suspicious, phishing and legitimate websites in real-time. Their

result showed that higher accuracy was achieved with two-fold cross-validation, i.e., a 98.5% true positive rate and a 1.5% false-positive rate. Moreover, this result demonstrated that the neuro-fuzzy system with five inputs was able to achieve higher accuracy in detecting phishing websites in real-time. The proposed solution (Barraclough *et al*., 2013) is right has it shows greater accuracy an improving efficiency of phishing website detection in real-time. The comparison mechanism was also better compared to another study. The proposed method could be improved further by adding more features and by optimising the parameters for greater accuracy. This approches is one of the study that that motive this study, because of the algorithm used, the feature selction and the level of accuracy derived in their experimnt.

Abdelhamid, Ayesh and Thabtah (2014) investigated the problem of phishing detection by using an associative classification (AC)[4] approach for data mining. They developed an AC algorithm called the multi-label classifier-based associative classification (MCAC)[5] and compared its performance with that of other AC rule induction algorithms when applied to phishing data. The experiment results showed that MCAC can improve the predictive precision of the AC algorithm which classify phishing website wrongly (Abdelhamid, Ayesh and Thabtah, 2014). Meanwhile, Xu, Wang and Jajodia (2014) proposed a simple but very efficient approach to prevent phishing attacks, which they called Gemini. The solution starts to work as soon as a user enters their username, and it tackles the phishing problem from a new standpoint. It is thus able to actively block access to phishing sites before the potential victim begins to enter their password (Xu, Wang and Jajodia, 2014). Chauhan and Shiwani (2014) proffered a solution based on honeypots being used currently in some solution. However, they proposed a solution to the limitation of the current solution by

[4] Associative classification (AC) is a data mining approach that uses association rule discovery methods to build classification systems (classifiers). [Online] Available at: http://www.worldscientific.com/doi/abs/10.1142/S0219649212500116?src=recsys& [Accessed 12 June 2016].

[5] Multi-label Classifier-based Associative Classification (MCAC) [Online] Available at: http://www.worldscientific.com/doi/abs/10.1142/S0129626414500017?journalCode=ppl [Accessed 12 June 2016].

performing the remodelling of the real online banking system into a considerable honeypot equipped with honeytokens (Chauhan and Shiwani, 2014).

The solution proposed by Gowtham and Krishnamurthi (2014) identifies phishing websites and notifies users that they are on such a website. However, their solution is solely based on the website login form, so it can be fooled or misclassify a site that does not have a login form (Gowtham and Krishnamurthi, 2014). Marchal *et al*. (2014) presented a solution called PhishStorm, an automated phishing system that analyses any URL in real-time to identify potential phishing threats. The result of their experiment showed that their approach gave 94.91% classification accuracy with only a 1.44% false-positive rate (Marchal *et al.,* 2014).

On the other hand, Singh, Jain and Maini (2015) proposed a useful scheme that is based on the pre-processing of classification algorithm and feature selection. The result of their experiment showed that RF was able to achieve the highest classification accuracy with 97.47% precision (Singh, Jain and Maini, 2015). Aydin and Baykal (2015) proposed a feature extraction technique for extracting website URL features by analysing subset-based features and classification algorithms for phishing website detection. The result of their experiment showed that the sequence minimal optimisation (SMO) algorithm had the best compatibility, achieving 95.39% accuracy, which was the highest accuracy obtained in their analysis (Aydin and Baykal, 2015). Feroz and Mengel (2015) proposed an approach that classifies URLs automatically based on their lexical and host-based features. Their model was able to detect a large number of phishing hosts with an average of over 93% accuracy (Feroz and Mengel, 2015). However, the above approaches chose to use website URL which classified as text content that can be improved with some of the other website features to improve detection accuracy according to the authors. Nanda and Gupta (2015) proposed a method of increasing security for online banking activities in order to reduce phishing attacks (Nanda and Gupta, 2015).

On the other hand, Dong *et al.* (2015) proposed a machine learning approach to detect phishing websites by using features from their X.509 public-key certificates. Thus, their approach works not only against HTTPS-enabled phishing attacks but also against HTTP phishing attacks. The result of their experiment showed that the RF algorithm was able to perform the best with a precision of 94.2% (Dong *et al*., 2015).

Jeeva and Rajsingh (2016) proposed a solution that is based on discerning the significant features that distinguish between phishing and legitimate URLs. The authors report that their model was able to detect 93% of phishing URLs using the rules obtained by the apriori algorithm (Jeeva and Rajsingh, 2016). Geng *et al*. (2016) also proposed an intelligent phishing detection system to address the phishing attack problem proactively. The authors report that their experimental results demonstrate the effectiveness and timeliness of the model in recognising phishing web pages (Geng *et al.*, 2016). They also demonstrated the efficiency of the model in distinguishing new phishing websites from real ones. Daeef *et al.* (2016) developed a detection system with a comprehensive level of protection by using URL features. Their system relies on the fact that users deal directly with URLs when surfing the Internet. The result of the experiment showed that the proposed system was able to achieve an accuracy rate of 93% by using URL detection (Daeef *et al.*, 2016). Therefore, the approach that the authors took to create the above solutions requires an additional feature and some elements of the website to make it more accurate.

On the other hand, Dadkhah, Shamshirband and Abdul Wahab (2016) proposed a hybrid approach based on the use of classification algorithms that are capable of identifying various types of the phishing site. The authors report that their techniques can identify periodical phishing attacks and legitimate sites that are embedded with malicious code more effectively compared to other techniques (Dadkhah, Shamshirband and Abdul Wahab, 2016). In a similar vein, Tahir *et al.* (2016) proposed a hybrid model for classification to overcome phishing website attacks. They tested various classification algorithms to determine which one would be the best to

incorporate into their hybrid solution. Moreover, their experiment result revealed that a combination of a Bayesian network (BN) and the instance-based learning with parameter k (IBK) model gave the best classification accuracy (97.75%) (Tahir *et al*., 2016).

Vargas *et al*. (2016) presented the results of an analysis into the elements of web pages that focused on the content and HTML structure as well as the domain registration records and DNS information of the sites in order to look for patterns and correlations between phishing sites. In this way, they provided insights into how phishers operate and provided guidance on how to build better tools for the detection of phishing activities (Vargas *et al.*, 2016)

Abutair and Belghith (2017) proposed a multi-agent system for phishing detection, which is an adaptive intelligent technique that acts on distributed case-based reasoning (CBR). The authors considered that a very significant advantage of their scheme is its ability to detect and avoid the zero-hour attack (Abutair and Belghith, 2017). It is also able to detect phishing attacks in a large-scale distributed system which shows the effectiveness and a high prediction. Weiss and Khoshgoftaar (2017) constructed several scenarios for phishing website detection. Besides, a novel transfer learning technique called canonical correlation analysis is used to align the feature space between the training and testing data. The authors report that the result of their experiment showed that their scheme was able to achieve an average of 87.5% accuracy (Weiss and Khoshgoftaar, 2017). Sonowal and Kuppusamy (2017) proposed an approach that detects phishing by using a multilayer model that contains five layers: an auto-upgrade whitelist, URL features layer, lexical signature layer, string-matching layer and accessibility score comparison layer (Sonowal and Kuppusamy, 2017). The experiment result showed that the proposed model was able to detect phishing sites with an accuracy of 92.72%. The detection rate of this approaches could be improved as the implementation is not best result and further work in this area is necessary such as adding more features because the number of features is small compared to the

dynamic way in which phishers use various tools to create a web page that is similar to real ones.

Park, Quadari and Tsang (2017) proposed a framework for phishing detection that discovers phishing websites based on existing and newly detected heuristics. The authors report that their heuristics-based solution is a new way of detecting phishing (Park, Quadari and Tsang, 2017). Abutair and Belghith (2017) proposed a case-based phishing detection system. Their scheme is designed to be updated frequently with approved phishing attack experiences. The authors report that their experiment result showed that their approach was able to achieve a classification accuracy of 95.62% (Abutair and Belghith, 2017). Subasi *et al*. (2017) proposed an intelligent phishing attack detection method that uses various data mining techniques to identify the classes of websites. The result of their experiment showed that among the classification techniques tested the RF algorithm gave the best performance, achieving an accuracy of 97.36% (Subasi *et al.*, 2017). The approaches are designed to protect user online against phishing attacks. Nevertheless, the solution was based on text-only and adding more website feature such as frame and image feature will improve the accuracy of the system.

Srinivasa Rao and Pais (2017) proposed a solution to automate the behavioural process of online users submitting fake credentials to the login page before submitting their actual credentials. Their application, FeedPhish, feeds a fake value into the login page. Based on their experiment results, the authors report that their proposed method was able to achieve an accuracy of 96.38%. Their application has an advantage in that it does not rely on any third-party service or prior knowledge such as the web history or a blacklist or whitelist of URLs. This service allows the scheme to detect not only zero-day phishing attacks but also phishing sites that are hosted on compromised domains (Srinivasa Rao and Pais, 2017).

Dhanalakshmi *et al.* (2017) proposed an end-host-based anti-phishing algorithm which is based on the characteristic of the phishing hyperlink. The approach that they used to develop their solution is an improvement on those proposed in some previous works, but it still requires additional features such as images and frames to make it more robust. Ramesh, Gupta and Gamya (2017) proposed a method that automatically identifies the victimised domain very effectively and differentiates domain with phishing web content. The authors report that the result of their experiments showed that their model is efficient in protecting users from phishing attacks with 99.54% accuracy (Ramesh, Gupta and Gamya, 2017). The developed methodology is an excellent way to start the detection of a phishing website. However, the small dataset means that the robustness of the scheme is not proven. They also need to incorporate more analysis into the scheme and expand the test.

Wu *et al.* (2017) proposed an innovative technique based on deep-learning techniques to address the challenges associated with phishing attacks. The results of their experiment show that the scheme is more effective and accurate (Wu *et al.*, 2017). In other previous work in this area, Shirazi, Haefner and Ray (2017) used an SVM in combination with stratified k-fold validation and grid search. The results of their experiment indicated that their system is more efficient and accurate than some comparable approaches. On the other hand, Machado and Gadge (2017) proposed an efficient way to detect phishing websites using the C4.5 decision tree approach and URL features. They reported that their solution provides 89.40% phishing detection accuracy (Machado and Gadge, 2017).

Wen, Zhao and Yan (2018) proposed a comprehensive associated analysis model for malicious web page detection that uses topic tracking, abnormal topic discovery, web page similarity, web page structure analysis and URL analysis. The authors reported that the result of their experiment showed that their method is better than the existing systems and has high classification accuracy (Wen, Zhao and Yan, 2018). Chin, Xiong and Hu (2018) proposed a scheme called PhishLimiter, which can be

described as a new recognition and mitigation approach, where deep packet inspection (DPI) is used while leveraging software-defined networking (SDN) to identify phishing activities through web-based and email communication. Their experiment showed that their model provides a practical solution to deter malicious activities (Chin, Xiong and Hu, 2018). The approach the authors adopted to develop their solution is better than those against which it was compared, but there is still room for improvement according to the author.

Gawade *et al*. (2018) proposed a solution in which they collected data from PhishTank for URL analysis and performed classification analysis on it using the RF algorithm. They develop an application that compares every parsed feature with a phishing feature, and if any features are detected, it would be classified as legitimate or phishing (Gawade *et al*., 2018). Oest *et al.* (2018) also proposed a new generic classification model for phishing URLs which applied up-to-date used social engineering techniques and reveals a relationship between URL type and compromised infrastructure use (Oest *et al.,* 2018). Thaker *et al*. (2018) as well proposed a system that can detect phishing URLs that have no past behaviours by which to detect them. The result of their experiment showed that their system was able to achieve an accuracy of 97.25% (Thaker *et al*., 2018). These approaches are mainly motivated to protect organisation interests. Nonetheless, they do not directly defend against phishing attack for users.

Babagoli, Aghababa and Solouk (2018) proposed a model for phishing website detection that utilises a meta-heuristic-based non-linear regression algorithm together with a feature selection approach. The authors report that their approach was able to produce a detection rate of 96.32% and performed better than an SVM (Babagoli, Aghababa and Solouk, 2018). Sankhyan *et al*. (2018) similarly proposed a solution that focuses on discerning the significant features that distinguish between fake and legitimate websites (Sankhyan *et al*., 2018). AlShboul *et al*. (2018) proposed a new

anti-phishing technique that can detect phishing websites and alert inexperienced Internet users to threats (AlShboul *et al*., 2018).

Parekh *et al*. (2018) proposed a solution for phishing detection that uses the RF algorithm to detect fake URLs. The authors concluded that their technique has an excellent performance in phishing detection with an accuracy rate of around 95%. Tyagi *et al*. (2018) proposed a solution for phishing prediction that is based on machine learning algorithms. For instance, in their experiment, the classification accuracy of the RF algorithm after applying PCA improved to 98.4% from 96.71% (Tyagi *et al*., 2018). Sonowal and Kuppusamy (2018) proposed a model based on a multidimensional similarity metrics scheme for screen reader users to help them detect phishing activities. Thus their work draws the attention of researchers to the need to develop anti-phishing tools to protect persons with visual impairment (Sonowal and Kuppusamy, 2018). Shyni, Sundar and Ebby (2018) proposed a technique called parse tree validation to detect whether a web page is phishing or legitimate. Their technique was able to achieve a false negative rate of 7.3% and a false positive rate of 5.2% (Shyni, Sundar and Ebby, 2018).

Jain and Gupta (2019) proposed an approach that can detect phishing attacks by analysing the links found in the HTML source code of the website. The solution incorporates many new outstanding hyperlinks specific features to detect a phishing attack. The result of their experiment was compared methods for phishing detection and discover that their scheme is relatively high accuracy in phishing website detection, which achieved more than 98.4% accuracy (Jain and Gupta, 2019). Recently, Li *et al*. (2019) presented a model to detect phishing web pages that use URL and HTML features. They also designed a lightweight features HTML and URL, which they introduced HTML string-embedding without using third-party services, which allows their model to work in a real-time detection application. The authors report that their scheme was able to achieve 97.30% accuracy and 4.46% true positive rate and 1.61% on a false negative rate (Li *et al*., 2019). Ulqinaku, Lain and Capkun (2019)

also proposed a solution called two-factor authentication (2FA)-PP; the scheme is a phishing detection that protects user's novel browser APIs that support direct communication between mobile devices and a web browser which enable the user's device to check the domain to which the user is accessing. The solution is can be integrated with some other 2-factor authentication models, such as QR codes and OTP that are an interactive and pairing device which are non-interactive (Ulqinaku, Lain and Capkun, 2019). However, the above model has impressive techniques an attacker can tamper with result of the verification using man in the middle attack and therefor corrupt the recognitions of all the system.

2.4.3 Blacklist-Based Approaches

Phishing attacks are widespread in today's cyber world, and they are increasing in number and complexity day by day. One of the other approaches that have been developed and implemented to resolve this issue is the provision of additional security features within Internet browsers that mostly rely on 'blacklisting', which is a process that compares a URL with a list of URLs belonging to the blacklist (Prakash *et al*., 2010). Blacklisting is an approach that is similar to the use of signatures by anti-virus programs that maintain a blacklist of sites that contain malicious content. Whenever a user tries to access a web page that is on a blacklist; an appropriate warning is generated to alert the user (Li *et al.*, 2014). Microsoft chose to use this approach and some heuristics in version 11 of IE (Microsoft, 2015). However, blacklisting is reactive and can be evaded by the rapid reusing of blocked phishing websites. Nevertheless, the blacklisting of known spammers has been one of the effective spam-filtering techniques. These methods may also contain domain used by known spammers, the IP addresses of open relays and proxies, virus and exploit attackers for better blacklisting solution (Chen *et al.*, 2014).

Whittaker, Ryner and Nazif (2010) developed a scalable machine learning classifier for detecting phishing websites. The classifier is used to maintain Google's phishing

blacklist automatically. Their classifier examines millions of web pages daily, analysing the URL and the content of the page to determine whether it is a phishing site. Their model classifies web pages that are submitted by end-users and those that are collected by Gmail's spam filters. The scheme also extracts and analyses some of the features associated with these sites. These features describe the composition of the web page's URL; the page HTML content and the hosting of the site that is collected by the web crawler. They trained their classifier on a noisy dataset consisting of millions of samples from previously collected live classification data. A logistic regression classifier was used to make the final decision as to whether a page was phishing or legitimate based on the selected features. Despite the noise in the training dataset, the classifier was able to learn a robust model for identifying phishing sites and was able to correctly classify more than 90% of the phishing pages (Whittaker, Ryner and Nazif, 2010).

Abraham and Raj (2014) proposed a string-matching method for detecting phishing attacks. Their method determines the degree of similarity a URL with blacklisted URLs. Hence, based on the textual elements of the URL, it can be classified as a phishing attack or otherwise. From their experiment results, the method was found to be effective in detecting phishing attacks with a shallow false-negative rate and an accuracy of 99.5% (Abraham and Raj, 2014). Hawanna, Kulkarni and Rane (2016) proposed a novel algorithm which detects whether a given URL is that of a legitimate or phishing website. Their algorithm performs a check against Google's updated blacklist and also utilises Google search engine results, the Alexa ranking and the number of URL-based features in order to detect phishing URLs. They use a safe browsing API to check URLs against Google's blacklist update for malware and phishing pages. The authors report that the algorithm is useful in detecting both known and unknown phishing URLs and that it can give a speedy response in the case of known phishing sites.

Furthermore, they stated that their solution provides the user with an alert message to let them know that the URL could lead to a potential phishing attack (Hawanna, Kulkarni and Rane, 2016). The approach used to develop the solution is decent, but it requires an additional feature to make it more efficient. However, they require an additional feature to make it more efficient in detection.

Li and Wang (2017) proposed a model called PhishBox that efficiently gathers phishing data and produces models for phishing validation and detection. The said approach incorporates phishing detection and validation and website data collection into an online tool that monitors the PhishTank blacklist to validate and detect phishing websites in real-time. Moreover, the model uses a two-stage detection procedure to ensure better performance. The result of the experiment showed that their two-stage model was capable of verifying phishing websites. Moreover, they monitored the blacklist and found that the blacklist contained a lot of invalid data. Hence, their scheme can remove five times more than regularly update in blacklist database after a week (Li and Wang, 2017).

Peng, Harris and Sawa (2018) proposed an approach that uses natural language processing techniques to analyse text and detect statements that are indicative of phishing attacks. Their approach is novel compared to the previous solutions because it focuses on the natural language text contained in the phishing attack and performs a semantic analysis of the text to detect phishing attacks. Natural language processing techniques are applied to parse each sentence and identify the semantic roles of essential words in the sentence. Their approach has a low false-negative rate, which shows that semantic information is a useful indicator in identifying phishing attacks (Peng, Harris and Sawa, 2018). They also report that the approach was able to achieve 95% classification accuracy.

2.4.4 Whitelist-Based Approaches

In phishing and pharming, the aim is to trick users into submitting their confidential information to fraudulent websites whose appearance looks similar to that of the original ones. The whitelist-based approach for phishing detection involves storing all the legitimate website URLs in a database such that any website visited by the user can be checked against this list to identify whether the site is phishing, suspicious or legitimate website. However, the whitelist approach is challenging to use because it is impossible to store all the legitimate websites that exist in the global cyber world. However, researchers have investigated the whitelist-based approach in order to protect the user from a phishing attack.

In light of the above, a proactive scheme to identify new phishing URLs accurately must be developed and implemented to improve the protection of online users (Le, Markopoulou and Faloutsos, 2010). Whitelist techniques are the most shared and straightforward type of anti-phishing solution. However, their general ineffectiveness means that this technology lags due to the continual emergence of new phishing websites (Huh and Kim, 2011).

Han *et al*. (2012) proposed an approach known as an automated individual whitelist (AILW) to protect the user's digital web identities. Their scheme leverages a naïve Bayesian algorithm to maintain an individual whitelist of a user automatically. If the user tries to submit his/her account information to a web page that does not match the whitelist, the model alerts the user about possible attacks. The scheme also keeps track of the features of login pages such as the paths to the DOM and the IP address in the input form. Their model can effectively protect users against phishing attacks and dynamic pharming. The result of their experiment confirmed that a model is a useful tool that can protect web identities (Han *et al.*, 2012). The detection algorithm can be improved through the addition of more techniques because the number of

features is small compared to the dynamic way in which features are used in phishing attacks.

For instance, Jain and Gupta (2016) proposed an approach to protect against phishing attacks that are based on the Google open DNS whitelist and hyperlink features and that use the auto-update of the legitimate site accessed by the user. Their approach can detect various types of the phishing attack, such as poisoning, DNS, embedded objects and zero-hour attacks in a real-time environment. The approach was able to achieve an 86.02% true positive rate and a 1.48% false-positive rate (Jain and Gupta, 2016a). Whitelist methods are the most common and straightforward solution. However, their ineffectiveness has made this technology lag behind new phishing websites.

Likewise, Armano, Marchal and Asokan (2016) proposed a new phishing detection technique that can be implanted as a client-side application and as a web browser add-on. Their application makes use of information that is extracted from a website visited by the user to detect whether the site is fake, and it warns the user if that is the case. In addition to detecting a phishing site, the solution also offers a redirect to the legitimate website. Their implementation was able to deliver the intended goal. The warning message produced by the warning system contains specific information about phishing attacks and offer the user three alternative options. First understanding the risk of a phishing attack by the user and second is to proceed to the website is the web page is in the whitelist, third is by clicking the check-in the warning message so as the application not to think the web page is phishing (Armano, Marchal and Asokan, 2016).

2.4.5 Toolbar Approach

Security organisations and experts and researchers in the field of security are concentrating on developing an anti-phishing toolbar which employs a whitelist-based, blacklist-based, content-based and heuristic-based approach in order to detect

phishing accurately and to prevent the user from becoming a victim of a phishing attack (Purkait, 2015). The approach proposed by Moghimi and Varjani (2016) aimed to determine the relationship between the page address and page content and was based on extracted feature sets. However, in addition, a hidden knowledge rule was formulated and embedded in a browser extension in order to detect phishing attacks in Internet banking sites, which achieved high accuracy and reliability (Moghimi and Varjani, 2016). The authors used an SVM algorithm to classify phishing web pages, and they used 10-fold cross-validation to test and train their model. The proposed model was able to detect phishing activities in Internet banking with an accuracy of 99.14% for true positives and 0.86% for false positives.

Sharma, Meenakshi and Bhatia (2017) carried out a survey and compared eight different types of phishing detection tool to find the tool that was the most effective. They tested each of the tools against a dataset consisting of 2,000 verified phishing websites reported between August 2016 and March 2017 that were collected from a reliable platform. The authors discovered that the anti-phishing toolbar performed very well as it was able to identify 94.32% of the phishing and legitimate websites in the dataset. However, they also conducted a survey and found that close to 61% of the respondents were unaware of phishing detection tools (Sharma, Meenakshi and Bhatia, 2017).

The toolbar developed in the current study uses a combination of the above methods. The publicly available information provided on the websites from which the toolbars were downloaded as well as our observations that were derived from using each toolbar gave us a basic understanding of how each toolbar works. An overview of some of the most well-known toolbars is provided below.

Cloudmark Anti-Phishing Toolbar:

The Cloudmark anti-phishing toolbar relies on the rating the user gives when he/she visits a website, where the user has the option of reporting that the site is good or bad.

The toolbar runs on Microsoft IE and displays a coloured icon for each web page visited. The toolbar shows a green icon for a site that is rated by users as legitimate and a red icon for sites that have been confirmed as fraudulent, while a yellow icon indicates that not enough information is available about the site to make a precise determination. Besides, users are rated according to their record of correctly identifying phishing sites (Wardman *et al.*, 2011). Furthermore, each site's rating is computed by aggregating all the ratings given for that site, while each user's rating of the site weighted according to that user's reputation.

EarthLink Anti-Phishing Toolbar:

The EarthLink Toolbar is a Firefox and an IE extension. It provides a collective user browsing experience with features to improve phishing detection. It also provides constant updates and pop-up blockers that prevent unwanted pop-ups when the user visits a website (Zhao *et al.*, 2017). Moreover, it also provides a spam blocker that warns the user about the potential risk of a phishing attack. Besides, it provides the location and further information about the organisation purported to be responsible for the website that the user is visiting (Arachchilage, Love and Beznosov, 2016).

Furthermore, it gives warnings to the user about the safety of websites by showing a red thumb sign for a phishing web page or a green thumb sign if the web page is safe, but if the web page is suspicious, it provides information about it (Arachchilage and Love, 2014). However, the toolbar does not provide real-time protection as the warning is provided after the user has input sensitive information into the web page, and the updating of the toolbar takes a long time.

eBay Account Guard Toolbar:

The eBay Account Guard toolbar provides an indicator that notifies the user that the current website that he/she is visiting is the bona fide eBay site. The green indicator indicates that the page belongs to eBay, the red indicates that the page is a known

phishing website that is on the blacklist database maintained by eBay, and the grey icon is for all other sites (eBay, 2010).

GeoTrustWatch Anti-Phishing Toolbar:

The GeoTrustWatch anti-phishing toolbar provides domain validation of the SSL certificate that employs encryption by signing the request form of the website through enrolment which is meant to protect the user from potential security risks (Upadhyaya, 2012). This toolbar function is useful against phishing because many legitimate websites use the SSL to encrypt the transmission of the user's sensitive information, whereas most phishing sites do not. Phisher avoids SSL because they need to obtain an SSL certificate from a public certificate authority (CA), such as VeriSign, which requires site identity information that can be traced back to the originator (Purkait, 2015). Using a CA that is not known to the browser will cause the user to be more cautious and thus increase their level of suspicious about a website. The solution also provides security for business identity authentication via secure 256-bit encryption, 2048-bit root and support for more than 99% of web browsers on both desktops and mobile devices. The toolbar only gives an identity of the user response for the less secure domain (Purkait, 2015). The toolbar provides the user with warnings in various colours and information about the website they are visiting, and it is free to download. As phishers use a range of deception tools, providing this type of warning information helps the user to check the legitimacy of the websites he/she visits.

GoldPhish Toolbar:

The GoldPhish toolbar was originally developed by Dunlop *et al*. (2010). It provides zero-day protection against phishing attacks with high precision. However, the time consumption of the system in scrutinising websites is quite high. The approach uses a browser plug-in to detect and report phishing sites. There are three main steps in GoldPhish that are followed. First, the current website in the user web browser is captured as an image. Next, OCR techniques are used to convert the image into

readable text. Then the text is input into a search engine to retrieve the results. Their application compares the top-level and second-level domain of the website the user is visiting with the first four in the Google search engine results. When a match is found, the application can verify the website and notify the user via the GoldPhish toolbar that the website is legitimate. The solution produced 98% true positives and 2% false positives when it was tested on over 100 sites. However, this method has unstable performance because an image could be manipulated with little variation, causing the process to fail in terms of accuracy. However, their solution is better than the previous image comparison methods that relied on a database. Nevertheless, more comprehensive tests need to be carried out as 100 sites are small in comparison to the number of phishing sites that spring up every day.

Google Safe Browsing Anti-Phishing Toolbar:

The Google Safe Browsing toolbar is an extension for Google Chrome and Firefox web browsers (Chiew *et al.,* 2015). This extension can alert the user that a site is fraudulent by referring to a blacklist when the user visits a web page (Chang *et al*., 2013). The extension uses two methods to detect a fraudulent web page. The user can either ask Google about each site he/she visits or download the Google list of websites that have been identified as suspicious. If the user downloads the website list, the web browser will update the blacklist each time before a new browser window is opened. The Google Safe Browsing extension check if the web page visited is in the blacklist stored locally (Google, 2016). However, if the user chooses to ask Google whether the website, he/she is visiting is legitimate or fake, the request is sent to a server maintained by Google. Then an analysis is conducted and returned by the server. If the page visited is considered to be a misleading one, the toolbar will stop the user's activities and give a warning advising the user on the next action to take, and the user is also able to report false or harmful web pages (Google, 2016).

Internet Explorer SmartScreen Filter:

Microsoft has incorporated a SmartScreen filter into its IE web browser that acts as a phishing filter. The filter utilises a server-side blacklist and a client-side whitelist that is maintained by Microsoft to perform a safety check to determine the legitimacy of a web page (Whittaker, Ryner and Nazif, 2010). If the web page is a phishing page, it automatically prevents the user from visiting the website (Microsoft, 2015). The detection mechanism uses the web address, which it sends to Microsoft's server to query it against the blacklist database and then returns the detection result to the client-side. This plug-in can alert the user about spoof web pages. After the user visits a website, the domain name is checked by the plug-in for the possibility of spoofing. Hence, the address that is visited is also checked against keywords that are stored in the blacklist. So, if the real domain name is different, the user is then alerted about a possible threat and the browser will block the user's attempt to visit the web page (Microsoft, 2015).

McAfee Anti-phishing Filter:

The McAfee anti-phishing filter checks the web page the user is visiting by providing the URL of the site to McAfee for evaluation and validation. If the web page is identified as a phishing page, a warning is presented to the user in a black colour and if the web page is suspicious it displays a warning in a grey colour, but if the web page is secure and safe the user is allowed to continue with his/her activities (Sharma, Meenakshi and Bhatia, 2017). However, the toolbar warning is passive, so some users may be unaware that this toolbar is installed on their browser (Armano, Marchal and Asokan, 2016).

Netcraft Anti-phishing Toolbar:

The Netcraft anti-phishing toolbar relies on a blacklist-based phishing filter that includes the URL, domain name, hostname, registration date and domain registration information (Li *et al.*, 2014). When a user browses the Internet, Netcraft checks the

URL and compares it with a blacklist stored in their server to process the information in the browser (Zhang *et al.*, 2012). This method uses various risk rates. If the web page the user is visiting has a risk rate that is higher than the prescribed rate for a safe website, Netcraft will inform the user about the danger and advise him/her on how to continue browsing safely. The main drawbacks of Netcraft are, firstly, that the connection with blacklist may be slow, so the warning pop-up takes time to appear, and secondly, the user may have to present their sensitive information to the web page before finding out whether it is safe or not. This means that the toolbar is both weak and time-consuming and therefore, may cause the user to be vulnerable to phishing attacks. The toolbar is designed for phishing prevention, even though most of its functionalities are not directly associated with the prevention of phishing. These are designed to identify fraudulent web pages they spoof, and they are short-lived against the legitimate site that is a US-based corporation but also registered in another country.

The heuristic-based anti-phishing technique uses website features such as text and frame content for phishing detection analysis to create a strong classification model (Lee and Park, 2016). Others use the blacklist/whitelist approach, which is similar to the use of signatures in anti-virus software solutions that maintain a blacklist of the sites that contain malicious content. Blacklisting is reactive and can be evaded by the rapid recycling of blocked phishing web pages. Therefore, for our solution, all the features and techniques listed in Table 2-1 will be explored in order to develop a robust phishing detection and protection scheme. As these features and techniques have not been used together in a single solution in any of the previous approaches, this represents the main strength of our advanced plug-in.

Table 2-1: Techniques Used by Anti-Phishing Plug-ins and their Level of Effectiveness

Anti-Phishing Plug-ins	Techniques	Browser	Effectiveness %	Service Type
GoldPhish	Heuristics & Features-based	IE	98	Free
Cloudmark	Heuristics	IE	94	Free
Microsoft SmartScreen	Blacklist and Whitelist	IE	95.9	Free
Netcraft (Customised)	Blacklist and Whitelist	Chrome; Firefox	90	Free
SpoofGuard	Heuristics & Features-based	IE	91	Free
Phishdentity	Google search-by-image API	IE	97.2	Research
PhisTackle	Heuristics & Features-based	IE	91.3	Research
PhishGuard	Heuristics	Firefox	94	Research
PhishIdentifier	Heuristics	Firefox	92	Research
PhishTester	Heuristics & Features-based	IE	97.1	Research
CANTINA+	Heuristics & Features-based	IE	98.06	Research
PhishAri	Features-based	Chrome	92.52	Research
PhishShield	Heuristics & Features-based	Chrome	96.57	Research
PhishNet	Blacklist	Chrome	95.0	Research
PhishDef	Heuristics & Features-based	Chrome	97	Research
Google safe browsing	Blacklist	Chrome; Firefox	93.3	Free
PhishZoo	Heuristics	Chrome	96.10	Research
Seclayer	Heuristics & Features-based	IE; Chrome	91	Free
IPDS	Heuristics, Features-based & Image-based	IE; Chrome; Firefox	98.55	Research

Table 2-2 shows the type of features used that each of the plug-ins listed in Table 2-1 uses for phishing detection. Column 1 in Table 2-2 contains the list of phishing plug-ins, and the rest of the columns illustrate the type of features each plug-in uses in their detection model. As indicated in the table, the majority of the plug-ins use text and a heuristic approach.

Table 2-2: Techniques and Features Used by Anti-Phishing Plug-ins for Phishing Detection

Phishing Plug-in	Techniques/Features					
	AI	**Frame**	**Heuristic**	**Image**	**Text**	**Whitelist & Blacklist**
GoldPhish			√	√	√	
Cloudmark			√		√	
Microsoft SmartScreen					√	√
Netcraft (Customisable)						√
SpoofGuard			√		√	
Phishdentity			√	√	√	

Phishing Plug-in	Techniques/Features					
PhisTackle				√	√	
PhishGuard			√		√	
PhishIdentifier			√		√	
PhishTester			√		√	
CANTINA+			√		√	
PhishAri					√	
PhishShield			√		√	
PhishNet						√
PhishDef			√			
Google safe browsing						√
PhishZoo			√		√	
Seclayer			√			
IPDPS	√	√	√	√	√	√

2.5 Adaptive Neuro-Fuzzy Inference System (ANFIS)

The ANFIS has been used for decades in engineering science to embed expert input into a computer model for a broad range of applications. It offers a capable alternative for determining operational risk. The integration of the neural network and fuzzy inference systems is formulated into concurrent and neuro-fuzzy models, which use human expertise by loading essential components in rule-based form and perform fuzzy reasoning to deduce the overall output value (Nguyen, Nguyen and To, 2016). There are two types of fuzzy inference system models: the Mamdani and the Sugeno model (Karaboga and Kaya, 2016). These fuzzy inference systems have two inputs and one output. Mamdani's fuzzy inference model was initially the standard fuzzy methodology and was the first control system built using fuzzy set theory. The Mamdani neuro-fuzzy system uses a supervised learning technique, namely, back-propagation learning, to acquire the parameters of the membership functions (Çakıt and Karwowski, 2017). On the other hand, the Sugeno neuro-fuzzy system applies hybrid techniques including back-propagation to study the membership functions, and least mean square estimation to fix the coefficients of the linear mixtures in the inferences rule (Pham *et al.*, 2018).

Aburrous *et al*. (2010) implemented an intelligent and efficient model based on an associative classification and data mining algorithm. This algorithm is used to identify rules and factors, to classify the phishing website and the relationship that correlate the factors and standards together. The authors demonstrated the flexibility of using associative classification techniques in an experiment involving a large dataset and showed that the proposed algorithm gave a better performance as compared to traditional classification algorithms. Nevertheless, they did not use different pruning techniques to remove rules that resulted in incorrect classifications, which reduced the accuracy rate of their algorithm (Aburrous *et al.*, 2010).

Ba Lam *et al.* (2014) also proposed techniques to detect phishing websites that apply fuzzy logic based on the features of the URL. They created an algorithm to check the URL of a page. The algorithm extracts some features in the URL, such as the primary domain, sub-domain, path domain and the domain itself (Ba Lam *et al*., 2014). The authors report that the result of their experiment showed that the approach had a 98.18% success rate in phishing site detection. Barraclough, Sexton and Aslam (2015) developed an online toolbar, which continuously runs in the background of the IE web browser, checking all websites user requests against a set of data in real-time. To detect phishing web pages, their approach uses a neuro-fuzzy scheme with six inputs: ethical site rules, user behaviour profile, PhishTank, user-specific sites, pop-up windows, and user credential profile. The toolbar was developed using 300 broad features based on six sets of inputs. This data is fed into the feature extractor algorithm based on neuro-fuzzy. The toolbar compares web page requests against features and downloads the website features if a suspicious site is detected (Barraclough, Sexton and Aslam, 2015). The result of their experiment indicated that their proposed approach improved the rate of phishing detection in real-time. However, they concentrated only on text-based features, so they could expand their approach further by analysing frame and image features to achieve better accuracy. The main contribution of their work is the introduction of a novel voice-generating user warning

interface algorithm for toolbar detection. This aspect of their toolbar will be integrated into our system implementation. Because of voice generation for visual impeded user and the colour warning system for those with hearing aids.

Paliwal, Anand and Khan (2016) developed an intelligent phishing system by using a fuzzy-based fuzzy inference system. They used a University College Irwin (UCI) machine learning dataset to test their scheme and found that the scheme gave satisfactory results. The developed model was tested against 60 websites, of which 30 were phishing websites, 15 were doubtful, and 15 were legitimate. Moreover, the dataset was tested on other classifiers to justify their claim. The authors report that their proposed approach was able to perform better than other classifiers such as naïve Bayes and the J48 system. The approach used to develop their solution is better than some others, but it requires additional features such frames and images, to make it more robust (Paliwal, Anand and Khan, 2016). Nguyen, Nguyen and To (2016) also proposed a solution for phishing detection based on neuro-fuzzy without using rule sets. Their approach has a high detection rate but fails to consider the valuable aspects of the website in the detection process (Nguyen, Nguyen and To, 2016). However, this limitation could be overcome by expanding the features used in the detection to include images and frames.

Pham *et al.* (2018) developed a neuro-fuzzy framework to detect phishing websites. The framework is based on URL features and web traffic features. Their designed scheme monitors phishing activities and protects fog users from phishing attacks. The framework consists of two components: an identification module and a backend. The identification module is deployed at the fog nodes to observe and detect the requested URL. This module discovers websites that pose a threat and prevents the user from visiting those websites. The backend is placed in the cloud and acts as a tool that manages the activities of the phishing attacks. The authors' experiment result showed that their model could effectively prevent phishing attacks in real-time at fog nodes and improve the security of the network (Pham *et al.*, 2018).

2.6 Deep Machine Learning

Currently, machine learning is continuously demonstrating its effectiveness in an extensive range of applications. This technology has come to the fore in recent times, owing to the advent of big data (Sahingoz *et al*., 2019). Big data has enabled machine learning algorithms to discover more fine-grained patterns and to make more accurate and timely predictions than ever before (Zhou *et al*., 2017). Machine learning techniques are used for object identification in images, the transcription of voice into text, matching news items and products with user interests and presenting relevant search results (Tyagi *et al*., 2018). The most common form of machine learning, whether deep or not, is supervised learning (Yao, Ding and Li, 2018).

Machine learning methods such deep learning (DL) have become a crucial tool for a broad range of applications such as image classification, natural language processing and speech recognition (Montavon, Samek and Müller, 2018). Machine learning is adopted in a wide range of domains, mostly in cybersecurity to evaluate the techniques to apply to the detection of intrusion, malware, spam and phishing (Apruzzese *et al*., 2018). Deep-learning architectures are composed of non-linear operations in multiple levels, such as neural networks with hidden layers, or of complicated relational methods in reusable approaches (Montavon, Samek and Müller, 2018). The deep-learning concept started with the study of artificial neural networks (ANNs) (Vazhayil, Vinayakumar and Soman, 2018), and it has become an active research area in recent years.

Deep-learning techniques have also been found to be suitable for big data analysis and been successfully applied in pattern recognition, computer vision, natural language processing, speech recognition and recommender systems. In a standard neural network (NN), neurons are used to produce real-value activations, and with the adjustment of weights, the scheme behaves as required. Moreover, training the ANN with backpropagation makes it useful with gradient descent algorithms which have

played a vital role in the model in the past decades. Although training accuracy is high with back-propagation, when it is applied to testing data, its performance might not be satisfactory (Liu *et a*l., 2017).

Yi *et al*. (2018) designed two sets of features for web-phishing interaction features and original content. They also developed a scheme based on a deep belief network (DBN). The test, which included using real IP flows from an Internet service provider (ISP), indicated that the proposed DBN-based model was able to achieve an approximately 90% true positive rate. Also, in the area automotive proposed in (CireşAn *et al*., 2012) in which a deep NN was used to assist the driver in the aspect of traffic light classification, the techniques were used to develop a system to assist in driving.

Below are some of the advantages of deep learning algorithms (Liu *et al.*, 2017):

1. It has robust unsupervised learning by getting most of its connecting structure in other to observe data, which is crucial in other to limit an enormous number of tasks and if the upcoming tasks are not known on time.
2. It can learn from mostly unlabelled data. This means that it can work in a semi-supervised situation, where not all of the dataset has comprehensive and correct semantic tags.
3. It can exploit the interactions that are existing across a vast number of tasks. These interactions exist because all that the algorithm task offer is a diverse view of the same underlying reality.
4. It can learn multifaceted, highly varying function with several disparities much higher than the number of training instances.
5. It can learn low-, mid- and high-level concepts with little human input, which is useful in terms of characterising the type of intricate functions that are required for deep-learning tasks.

6. It can learn from a massive dataset of features and can compute the training data in a short period with several linear examples.

However, there are some challenges associated with deep learning algorithms regarding the issue of the data used (Guo *et al.*, 2016), as follows:

1. **Unbalanced data**: This is an issue that occurs in learning and mostly happens during classification if there are more features of some class than others. This issue can be resolved by using some techniques that focus on the data level or the classifier level.
2. **Inadequate data for learning**: This is an issue that occurs when a limited amount of data is available for cross-validation methods which are mostly applied by dividing the available data into two sets, one for learning and the other for validation, in order to check the behaviour of the network. However, to gain a better knowledge of the network, the size and features may be modified for training and evaluating the various aspects of the network.
3. **Overflow of data**: This problem occurs in big data because the generation of data is growing exponentially, and it is forecast that the information contained big data will continue to increase daily.
4. **Partial data**: Sometimes, a collection of data is used for solving a particular task, but the data becomes partial when some of it is lost or because some of its variables or features are unidentified. To resolve this issue, it is necessary to approximate missing values and then discover the relationship between the identified and unidentified data. There some methods based on NNs (Liu *et al*., 2017) and some other approaches that can be used to solve the problem.
5. **High-measurement:** Information in the real-world application is often overflowing from the determination of a specific problem point of view which can be handled by the algorithm.

2.6.1 Long Short-Term Memory (LSTM)

Long short-term memory is based on the recurrent neural network (RNN), which is used to recognise the occurrence of patterns in time series and which also uses error flow in its analysis. However, the LSTM architecture was developed to overcome the shortfalls in RNN, which is a highly non-linear recurrent network with multiple gates and propagative feedback (Breuel *et al*., 2013). An LSTM layer contains a set of recurrently connected blocks, known as memory blocks. These blocks can be a look-alike version of memory chips in a digital system. Hence, each of the blocks includes one or more repeatedly connected memory cells and contains three multiplicative units, namely, the input, forget gate and the output, which provide non-stop analogues of the read, write and reset functions for the block cells (Sundermeyer, Schlüter and Ney, 2012). The LSTM network has achieved excellent results in character recognition applications (Breuel *et al*., 2013). It has also been used extensively in the analysis of handwriting recognition, speech recognition and polyphonic music modelling, where the results have shown that its usage leads to an improvement in standard detection analysis with variance in the parameter (Greff *et al.*, 2017). It has also been used in language modelling to analyse speech in a speech recognition system, where it was found to show an improvement in confusion over the RNN (Sundermeyer, Schlüter and Ney, 2012).

Bahnsen *et al. (*2017) investigated the performance of LSTM in their work on a solution for phishing site prediction that uses URLs as input for machine learning models. The authors compared a feature engineering approach with random forests (RF) classifier against a novel method based on RNNs. They used 14 features to build their lexical and statistical analysis of the URLs. They used an LSTM unit to build the model that receives as input a URL as a sequence of character and predicts whether the URL is phishing or legitimate. They also constructed a dataset that consisted of two million phishing and legitimate URLs to train their model. They found that the LSTM

model had an overall higher prediction accuracy compared to the RF classifier without the need for expert knowledge to create the features. Their approach was able to achieve an accuracy of 98.7% even without the need for manual feature creation (Bahnsen *et al*., 2017). Their approach was part of the study that motivate this study using deep learning algorithm, because of the level of dataset used in their experiment and the features used with the result.

2.6.2 Convolutional Neural Network (CNN)

In recent years, the convolutional neural network (CNN) has seen massive adoption in computer vision applications (Yu *et al.*, 2017). In the area of object recognition, CNN has also been used for feature extraction (Xu, Li and Deng, 2015). The CNN belongs to the family of multilayer NNs that are developed for use with two-dimensional data, such as videos and images (Arel, Rose and Karnowski, 2010). CNN is one of the most prominent deep-learning methods where numerous layers are trained using a rigorous methodology.

Recently, Yang, Zhao and Zeng (2019) proposed an approach for a multidimensional feature phishing detection solution that is based on a fast classification method using deep learning. In the initial stage, they extract the feature and sequence character of the URL and use deep learning for quick classification; note that this step does not require third-party assistance or prior awareness of phishing websites. In the next stage, they combine the URL demographic features, web page text, code features and the quick CNN-LSTM classification into multidimensional features. In total, they extracted 24 features from a given website to develop their model. After the extraction of the features from the different elements of the website were fused, they were fed in the CNN-LSTM algorithm is to generate an output used as in-depth URL features and combine it with the statistical URL, web page code, and the text features to build the multidimensional features that are classified using a machine learning approach. According to the authors, their approach reduces the

detection time for setting a threshold that reduces the classification time. The scheme was applied to a dataset containing millions of legitimate and phishing URLs. The result of their experiment showed that it was able to achieve an accuracy of 98.99% and a false positive rate of just 0.5% (Yang, Zhao and Zeng, 2019).

Likewise, Vazhayil, Vinayakumar and Soman (2018) performed logistic regression using CNN, CNN-LSTM and a bigram to evaluate two datasets of URLs for phishing detection. They collected the dataset from four different sources: the MalwareDomainlist and MalwareDomain for malware URLs, and PhishTank and OpenPhish for phishing URLs. The dataset contained 60,000 training URLs and over 56,000 testing URLs. The dataset was used to train the CNN and CNN-LSTM models to detect phishing URLs. The choice of the algorithm is that it can take raw data URLs as their input. The result of the experiment showed that the CNN-LSTM architecture performed better than the other model, achieving an accuracy rate of about 98% for the classification of URLs (Vazhayil, Vinayakumar and Soman, 2018).

Similarly, Yao, Ding and Li (2018) proposed a detection method with fast object recognition techniques using an improved R-CNN for small-scale identification. They decided to use a faster R-CNN with a feature pyramid network (FPN) for logo recognition because of the limited size of the two-dimensional code and because the size of logos embedded into websites is also small. Their method is comprised of three processes: recognition and extraction, logo extraction, and recognition and identification, First, the logo extraction process extracts the image used as a logo on a website as a two-dimensional code. Next, the recognition process is improved by using a fast R-CNN to identify the logo. The final process is the identification, which evaluates the consistency between the true uniqueness of the requested website and its defined identity. If the given website logo has consistency, then the site is confirmed as legitimate, whereas if it has inconsistency, then the website is considered to be a phishing site. The scheme was implemented on the Flickr logo-32plus dataset. The authors report that the result of their experiment showed that their proposed method

was able to perform logo recognition more effectively than other methods (Yao, Ding and Li, 2018).

Yuan *et al*. (2018) also proposed an in-depth learning approach for phishing detection using URL characters. They mapped URLs to documents and words using a word2vec-based embedding learning method. Hence, they combined the structure of the URLs with the embedded characters to acquire a vector representation of the URLs. Their detection system consists of three modules. One is the character embedding learning module which stores the vector representation of the characters in the URLs. Part of the detector module which is the third that involves training algorithms on the vector illustration of the URLs to classify them into legitimate and phishing websites. The scheme was tested on a publicly available dataset containing one million phishing websites. According to the authors, the result of the experiment showed that their method was able to achieve an accuracy of 99.69% (Yuan *et al*., 2018).

Similarly, Le *et al*. (2018) proposed a solution called URLNet, which is an end-to-end deep-learning framework for learning non-linear malicious URLs by detecting it from the URL. They applied a CNN to both the words and characters of the URL features to learn the URL embedding in a jointly optimised framework. This approach allowed their model to capture several types of semantic data, which would not have been possible using existing schemes. They also presented advanced word-embeddings to solve the problem of too many rare words being observed in a classification task (Le *et al.*, 2018). They conducted their experiments on a large-scale dataset and demonstrated that their proposed method gave a strong performance that was better than that of an existing method. The approach has two branches; the first branch has a character-level CNN where character-level embedding is used to represent the URL. The second branch contains a word-level CNN where word-level embedding is used to represent the URL. Thus, word-embedding itself is a mixture of

character-level embedding and individual word-embedding. Their approach works in such a manner that it does not require any expertise.

As mentioned above, CNN has also been shown to be highly effective in computer vision applications (Guo *et al*., 2016) and is, therefore, commonly used for that purpose. The CNN contains an input layer, convolution layer, pooling layer, fully connected layer, and output layer. The input layer holds the raw image values; the convolutional layer computes the output of the node that is connected to local regions in the input layer; the pooling layer performs a down-sampling process along the three-dimensional dimensions; the fully connected layer calculates the session scores, and the output layer produces the results. Currently, three main techniques are used in CNN for image classification: (1) unsupervised pre-training of the CNN with supervised fine-tuning, (2) transfer learning by fine-tuning the CNN models that have been pre-trained on a natural image dataset and (3) training the CNN from scratch using available pre-trained features (Yao, Ding and Li, 2018).

Recently, Li, Wang and Kot (2017) proposed using the RNN and the CNN for image recapture detection in order to learn the deep representation of the images in order to extract discriminative and essential features of the intra-block and inter-block information of images (Li, Wang and Kot, 2017). Also, LSTM and CNN were used in combination by (Xu, Li and Deng, 2015) to learn the temporal structure of video in order to show how the temporal features are used for face anti-spoofing purposes and to differentiate the genuine attempt to identify a fake website.

2.7 Non-Technical Anti-Phishing Solutions

Non-technical anti-phishing solutions are also available and can be classified into two types: legislation and user awareness. Legislation or Acts have been passed all around the world to criminalise the activity of creating phishing websites. User awareness is

promoted by educating end-users about phishing to help them to recognise phishing websites and how to avoid them (Shabut, Lwin and Hossain, 2016).

2.7.1 Legislative Tools

The use of legislation is a direct measure to reduce phishing by tracking and arresting those who are involved in this criminal activity. The US was the first nation to use laws to combat illegal cyber activities, and many cyber attackers have been arrested and arraigned. The main issue with this approach is the effectiveness of the laws as it is challenging to trace phishing attacks. Fraudulent websites naturally migrate quickly from one server to another. Also, an average phishing website is online for less than 48 hours (Oest *et al*., 2018). Hence phishing attacks are committed very quickly and, subsequently, the criminals who commit these attacks also quickly disappear into cyberspace. The other issue is that many laws are applied only when the damage has been done, and the online user has already been defrauded as a result of phishing attacks.

2.7.2 User Awareness

Purkait, Kumar De and Suar (2014) presented a report on the result of an empirical investigation into the various factors that have a significant effect on the Internet user's ability to identify a phishing website. In their empirical analysis, they used some groups of Internet users who had at least some experience of financial transactions over the Internet. They conducted quantitative research with the help of a structured survey questionnaire and also performed three experimental tasks. A total of 621 sound samples were collected, and multiple regression analysis techniques were used to deduce the answer to the research question. The result of their study showed that their model was useful and had an explanatory influence. Their analysis also showed that 92.7% of the Internet users could identify a phishing website but cannot explain by the prediction selected for the model (Purkait, Kumar De and Suar, 2014).

Nevertheless, educating users remains a critical aspect of phishing detection because users need to be aware of phishing techniques and of how reputable organisations would communicate with them on the web and via email; the lack of phishing education among users is one of the contributory factors to phishing attack success. Due to the growth in cyberspace technology, computer users have a significant role to play in making the Internet a safer place for everyone because cyber attacks are targeted at achieving either financial or social gain (Arachchilage and Love, 2014) to the detriment of the user. On the other hand, some people undertake phishing activities for fun and a sense of accomplishment rather than for financial or social gain.

Phishing awareness has been improved through the development and use of online game training and email-based training to combat phishing attacks. However, there will always be some inexperienced users accessing Internet web browsers, which can quickly become phishing targets. Moreover, phishing techniques are continually being improved to such an extent that even experienced Internet users can still be fooled by phishing websites (Shabut, Lwin and Hossain, 2016). Thus, it is challenging to combat phishing solely through education because not only do users not read the educational materials; it is hard to teach users how to make the right decision online. Therefore, continued user training and awareness may be the key to combating phishing attacks in organisations (Jansson and von Solms, 2013).

2.8 Organisational Best Practice to Combat Cyber Attacks

The best practice that an organisation can follow in order to handle cyber attacks and protect against them is to create a policy framework that guides all of the staff of the organisation in their day-to-day activities against privacy and security. Mouratidis *et al.* (2012) proposed a framework that supports the unified analysis of security and privacy. The scheme uses a meta-model that combines concept from privacy and security requirement methods. The methods include privacy and security goals, constraints, properties, process pattern and actor within a social framework. The proposed

framework provides a holistic approach overcoming the drawback information system security by analysing security and privacy from the requirement engineering stage. The framework also provides a unified requirement that is based on organisation views an actor which suggest the correct implementation procedures for the respective privacy and security which provides a solution to successfully bridge the gap between the requirement and implementation stages (Mouratidis *et al.*, 2012). Alkhozae and Batarfi (2011) proposed a solution for securing web system that is based on quality models used in security requirement engineering method (Alkhozae and Batarfi, 2011). The scheme provides a means for refinement and requirement repository for future reuse.

Below are some steps that need to be followed to achieve this aim in protecting an organisation against a cyberattack:

- **Create company policies and communicate them to users**: Create organisational policies for email content so that legitimate websites cannot be confused with phishing sites. These policies should be created and sent to users with proper monitoring. The organisation should also carefully assess the effect of such a policy on the user experience versus the increased security provided by implementing the policy (Wang, Kannan and Ulmer, 2013).
- **Implement stronger authentication of websites**: The organisation should create a stronger authentication mechanism and should not ask users for sensitive information when they log on to the organisation's website. This will make it difficult for a phisher to extract such information from the user (Xinming, Jing and Jun, 2010).
- **Offer measures for the user to validate the legitimacy of an email**: The organisation should embed authentication information into every email that is sent to users, and they should be able to identify that the email is from the said organisation and not from a phisher (Cox, 2012).
- **Monitor the Internet for possible phishing websites**: A phishing web page generally appears somewhere on the Internet before the launch of the phishing

attack. Also, most of the sites sometimes modify the organisation's trademark to appear legitimate (Chiew *et al*., 2015). The organisation should, therefore, monitor the Internet for sites that look similar to its legitimate site.

- **Implement good-quality anti-spam, anti-virus and content filtering solutions at the Internet gateway**: The intrusion detection and firewall system at the Internet gateway provides a scanning functionality and acts as an additional layer of defence for the internal network. This type of system can block new phishing sites at the gateway. A gateway anti-spam-filtering system can help users to avoid unwanted spam and phishing websites (Chen *et al*., 2014).

2.8.1 User Best Practices

An organisation needs to secure its environment against phishing attacks and reduce its vulnerability, but it also needs to educate users on how to approach any suspicious email activities on their system and set up an automatic blocking mechanism to protect itself and its users from some known malicious sources. The organisation can do this by:

- **Automatically blocking malicious or fraudulent email**: Spam filtering can assist in keeping the user from ever opening a suspicious phishing email, but spam filtering is not foolproof (Inuwa-Dutse, Liptrott and Korkontzelos, 2018).
- **Automatically detecting and deleting malicious software**: A phishing attack often make spyware tools for attacking users when they find their way into the user's computer (Arachchilage and Love, 2013), but many commercial programs can remove this type of software, which is the focus of this research in our future development.
- **Automatically blocking outgoing delivery of sensitive information to malicious parties**: Although some users may not be able to visually identify whether a website that requests sensitive information is legitimate, which is the

strength of our solution is to support the user in an automated form (Cohen *et al.*, 2016).

- **Automatically blocking outgoing delivery of sensitive information to malicious parties**: Although some users may not be able to visually identify whether a website that requests sensitive information is legitimate, which is the strength of our solution is to support the user in an automated form (Cohen *et al.*, 2016).

2.9 Chapter Summary

This chapter provided an in-depth overview of phishing activities and methods of phishing detection. The key approaches, results, strengths and weaknesses are identified with relevant implications for this work highlighted. More generally, it still suffer a bit from from beign too ong and not begin able to see the wood for the trees, but is definitely now less repitation.

CHAPTER 3: RESEARCH DESIGN AND METHODOLOGY

Chapter 3 Research Design and Methodology

3.1 Introduction

A methodology in research is a systematic approach to extract features from a given sample to understand a given problem and various aspects of the research (Kumar, 2013). For this study qualitative research is used to collect non-statistical features whilst a quantitative research approach is used to analyse numerical features and experimental results. This chapter will discuss the methods adopted in this study, including phishing techniques, feature extraction methods, intelligent systems, the knowledge model, the Adaptive Neuro Inference System algorithm (ANFIS), and phishing detection and web browser plug-in modelling. The chapter will also present details on the chosen website feature extraction methods and their reliability and accuracy, and also evaluate current knowledge models and phishing detection methods. In this approach, the process for identifying phishing website is illustrated in Fig. 3-1, the parameters utilised ANFIS by using features to optimise phishing website detection accuracy and minimise error rates with the assumption that a proper parameter tuning framework based on ANFIS using full features will enhance phishing detection accuracy in real-time. In the study, a web browser plugin for phishing detection will be developed that has a user warning interface that enhances the alerts given to users to ensure that they are effective in real-time. A toolbar will be implemented in MATLAB version 9.5 AppDesigner with a voice warning interface and text directives and other extensive features to improve phishing detection.

The vast progress in recent time that has been made in the development of data mining techniques has led to an extensive variety of algorithms that have been applied in the field of statistics, machine learning, pattern recognition and features extraction to form datasets for phishing detection (Zhang and Yuan, 2013). The relationships between association rules and classification rule algorithms can identify features or characteristics that can be used to describe a dataset. Once a phishing website and

the vital features and factors that are inherent to it are identified the next step is to stipulate how the different elements of the phishing website are related to one another. This is done using fuzzy rules in the form of ***if…then*** statements that relay phishing website possibility to various levels of main phishing features based on the experience and knowledge of the software security developer. However, in this study, instead of just employing an expert system, a machine learning approach is taken utilising a fuzzy inference system (FIS) which is optimised with ANFIS to generate a new phishing website detection scheme.

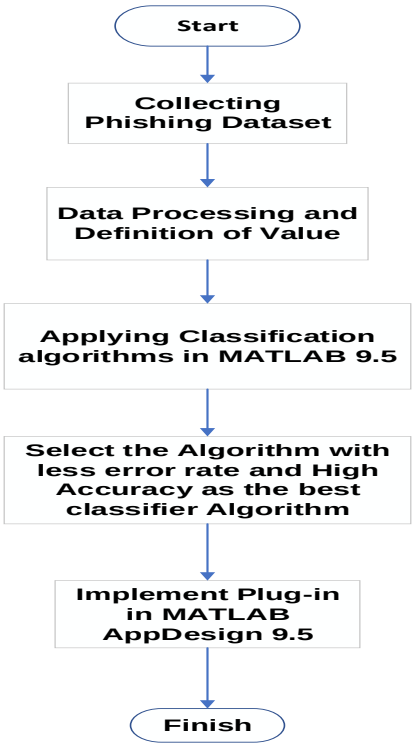


Fig. 3-1: Methodology for Identifying Phishing Website

The scheme was used to automatically find the essential features that are related and the associations between different patterns of phishing features in the phishing website collection dataset. Several different machine learning and classification techniques were used and implemented within MATLAB version 9.5, which includes a classification toolbox containing several different methods that can be applied to many datasets such as association, classification and regression. Support Vector Machine

(SVM) and K-Nearest Neighbour (KNN) classification algorithms are used to discover the relationships between the selected phishing elements. For the ANFIS algorithm, the neuro-fuzzy design toolbox was used as provided by Barraclough, Sexton and Aslam (2015). These classification algorithms were chosen because of the diverse approaches they use to create rules, and because their learned classifiers are easily understood (Çakıt and Karwowski, 2017).

3.2 Source Selection and Their Patterns

The current study builds on the previous work of (Barraclough, Sexton and Aslam, 2015) and (Barraclough *et al.*, 2013). For this present research three sets of website features were identified as sources and were processed in two phases. The first phase consisted of extracting 300 features from April 2016 until December 2017. This data comprised text and frame. The list of features later prunes down to 30 after identifying those that are best used for phishing classification. This was done by eliminating redundant features as identified in previous work (Barraclough, Sexton and Aslam, 2015). The second phase increases the number of essential features to 35 using additional source (image) data, extracted from November 2016 to February 2017, hence makes three sets of input sources in total. The three sources are sub-divided as follows; the search index has four elements, the URL content has six parts, there are five elements for the web address bar and image identity, four elements for the domain identity, three features for the source code & JavaScript, and eight features for the page style and layout identity. The list is presented in Appendix A.

3.2.1 Feature Extraction

The purpose of the feature extraction process employed in this work is to extract website features in order to reduce the unique feature space for phishing detection. In other words, the unique features are reserved and changed into a new compact space with fewer representative sets (Zareapoor and Seeja, 2015). This method primarily

uses principal component analysis (PCA) and latent semantic analysis (LSA). The PCA technique decreases the size of the data by changing the actual feature space into a smaller one (Vidal, Ma and Sastry, 2016). This is achieved by converting the real variables $Y = [\mathrm{y1}, \mathrm{y2}, \ldots, \mathrm{yn}]$(where n is the number of the actual variables) into a new set of variables $T = [\mathrm{t1}, \mathrm{t2}, \ldots, \mathrm{tp}]$ (where p is the number of the new set of variables). The LSA technique is an innovation that is based on text classification. This approach analyses the relationship between a concept and a term contained in unstructured data, and it can correlate semantically related terms that are latent (Marcolin and Becker, 2016). These feature selection processes were followed (code in Appendix C) in order to find better features for more intelligent phishing detection in a dynamic approach that can achieve high true positive (TP) and low false negative (FN) results.

The features example use in the present study is the complete source from which website features were gathered for this study. For the identification of the features, journals and reported cases were carefully selected. Phishing website and legitimate website features were randomly selected to derive an unbiased representation of the large number of sources available. An account registration with PhishTank was completed to allow access to various information and tools, including archive websites that are maintained by community volunteers (PhishTank.com, 2012). The phishing websites from PhishTank consist of verified phishing and unknown phishing websites submitted within three years from 1st of June 2014 to 30th June 2016. The website archive maintained a total of 1,500.456 verified phishing sites by the middle of 2014. The chosen legitimate website source was randomly chosen, namely the Financial Service Authority (FSA) website, using search keywords. The FSA is a UK financial service institution known by users to be a legitimate website. The WHOIS website was also used as a source for a legitimate website. This website was accessed between April 2016 to June 2018 using search keywords, and whose features were used to differentiate against phishing websites.

The three sets of inputs including text, image and frame were chosen randomly and combined as one feature called the 'hybrid features' in this study. Of the over 290 features that were collected from phishing websites, 35 of these features were used for experimental purposes to analyse the performance of our ANFIS-based phishing detection method. These 35 elements can help to distinguish between legitimate and phishing websites. The three approaches are described in the following sections 3.2.2 to 3.2.4.

3.2.2 Text-based Features Extracting Approach

The text-based approach depends solely on website content such as the search index, security & encryption, web address bar, domain identity and the source code to detect phishing sites. It can detect new phishing websites that are not yet blacklisted and that are targeted at unsuspecting users. The text features are related information available to the user and some key identifier of the websites. This exploration was conducted during the period of 2nd April 2016 to 30th September 2016.

Search index features:

- **Page ranking**: This feature is used to check the importance of the web page. By counting the number of quality links to a page, this can determine the importance of the site on the Internet.

  The rule below was created to check whether the domain part of the website is among the top addresses listed on a Google search engine. If the site is not among these, then it is considered a phishing site, but if it is among them, it is judged to be a legitimate site.

$$\text{Rule: IF} \begin{cases} \text{PageRank} < 0.2 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Google index:** This feature is used to ascertain whether the URL of the website in the Google index matches the one submitted to Google index.

The rule below is created to check whether the domain part of the website is listed on the Google index. If the site is not in the index, then it is considered to be a phishing site. Otherwise, it is a legitimate site

$$\text{Rule: IF} \begin{cases} \text{Web page Indexed by Google} \rightarrow \text{Legitimate} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}.$$

- **Website traffic:** This feature is used to measure the amount of data sent to and received by a visitor to a website.

  The following rule is created to check the amount of data that is sent to and received from another website. According to Rami, McCluskey and Fadi (2015) if it is less than 100,000, the site is considered legitimate, but if the number is higher than 100,000, the website is considered to be suspicious otherwise, if it is 130,000 and above is considered a phishing site.

$$\text{Rule: IF} \begin{cases} \text{Website Rank} < 100{,}000 \rightarrow \text{Legitimate} \\ \text{Website Rank} > 100{,}000 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phish} \end{cases}$$

- **Statistical report**: This feature is used to ascertain the usage of the website, such as the number of queries and website availabilities online. However, a new website may fail this check, so some other features are used to ascertain the legitimacy of such websites.

$$\text{Rule: IF} \begin{cases} \text{Host Belongs to Top Phishing IPs or Top Phishing Domains} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

Security & encryption features:

- **Existence of "https" in URL:** This feature is used to check whether the URL contains the https as most phishers use this to deceive the unsuspecting user. A website with a URL containing https after the top-level domain is considered to be a phishing site as it is rare that the https appears after the top-level domain in legitimate websites because one of the methods to conceal the destination

is to use another website domain inside the actual link of the website (Google, 2016).

The following rule is created to check whether the linked part of the website contains https after top-level domain; if it does contain the https, the site is considered phishing; otherwise it is judged to be a legitimate site.

For example, the URL has https address: https://www.yahoo.com is more dependable than this phishing address http://ebay.com/https/santos.com/gucci2014/gdocs/gucci.php?Acirc=A?A?=A?Auffe0

$$\textit{Rule}: \text{IF}\begin{cases}\text{If The link Part has an https} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Long URL**: This feature is used to check the length of the URL to determine whether the original website has the corresponding URL. A reliable and trustworthy website will always come with a short URL address than the one with a long URL address (Tan, Chiew and Sze, 2014). For example, the URL with a short address: http://www.yahoo.com is more dependable than this suspicious address: http://unitedstatesreferral.com/santos/gucci2014/gdocs/gucci.php?Acirc=A?A?=A?Auffe0

The rule below is created to check the legitimacy of the URL, where a URL with a length of fewer than 54 features is likely to legitimate, a URL with 55 to 75 features is deemed suspicious, and a URL with more than 75 is considered to be a phishing site.

$$\textit{Rule: IF}\begin{cases} URL\ length < 54 \rightarrow feature = \text{Legitimate} \\ else\ if\ URL\ length \geq 54\ and\ \leq 75 \rightarrow feature = Suspicious \\ otherwise \rightarrow feature = \text{Phishing}\end{cases}$$

- **Use of the IP address**: This feature is used to check whether the URL contains the IP address as most phishers use this to deceive the unsuspecting user. A website with a URL containing an IP address is considered to be a phishing

site as it is rare that the IP address appears in legitimate websites because one of the methods to conceal the destination is to use the IP address of the website.

The following rule is created to check whether the domain part of the website contains IP address; if it does contain the IP address, the site is considered phishing. Otherwise it is judged to be a legitimate site.

$$\underline{Rule}: \text{IF} \begin{cases} \text{If The Domain Part has an IP Address} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Abnormal URL**: This feature can be used to check the abnormalities in the URL against the information stored in the WHOIS database for legitimate websites. Then determine if the hostname in the URL matches the claimed identity as a URL is unique on the Internet for a legitimate website; its identity is usually part of its URL.

  The following rule is created to ascertain whether the URL is abnormal or not. It specifies that if the hostname does not correspond to the URL, the site is a phishing site, else it is legitimate.

$$\text{Rule: IF} \begin{cases} \text{The Host Name Is Not Included In URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Abnormal request:** This feature is used to determine whether there is a request from an external object within the web page, such as an image or video that has been loaded from another domain. In the case of a legitimate, official website, a considerable percentage of those URL is in its original domain.

  The rule below is created to check the percentage of the URL that is loaded from another website. If it is less than 22%, the site is considered legitimate; if the percentage is higher than 22% but less than 61%, the website is deemed to be suspicious, else it is considered to be a phishing site.

$$\text{Rule: IF} \begin{cases} \%\ \text{of Request URL} < 22\% \rightarrow \text{Legitimate} \\ \%\text{of Request URL} \geq 22\% \text{ and } 61\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{feature} = \text{Phishing} \end{cases}$$

- **Abnormal anchor:** This feature is used to check whether the anchor element is like a tag <a> from an external link. This feature is treated as the request URL. A web page is suspicious when the domain of most of the URL looks abnormal compared to the domain of the page, and the anchor does not link to any page.

  The following rule is created to check the percentage of the URL that is loaded from another website. If it is less than 31%, the site is considered legitimate; if the percentage is higher than 31% but less than 67%, the website is seen as suspicious, else it is considered to be a phishing site.

$$\underline{\textit{Rule}}\text{: IF} \begin{cases} \%\ \text{of URL Of Anchor} < 31\% \rightarrow \textit{Legitimate} \\ \%\ \text{of URL Of Anchor} \geq 31\% \text{ And} \leq 67\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

Web address bar features:

- **URL contains a prefix or suffix**: This feature is used to check whether the dash symbol that is rarely used is in a valid URL. Phishers tend to add a suffix or prefix to the domain name that is separated by a dash (-) to make users feel that they are dealing with a legitimate web page, for example, http://www.online-paypal.com or http://www.barclaybank-card.com. From these examples, we can see that the word 'online,' which appears as a prefix before the legitimate PayPal domain name https://www.paypal.com is done to confuse the user.

  The rule below is developed to check for the prefix (-) in a web page URL to determine whether the link contains the prefix.

$$\text{Rule: IF} \begin{cases} \text{Domain Name Part Includes } (-) \text{ Symbol} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **URL contains a '@' symbol**: This feature is used to check for the @ symbol in the URL as it leads the browser to ignore everything preceding the @ symbol. In this context, if the format <userdata>@<hostdomain> is used, the browser ignores the <userdata> and directs the user to <hostdomain>. To further hide the URL, the @ symbol can be represented by its hexadecimal character code "%40".

  The rule below checks for the presence of @ symbol in a given URL, and if it is present, the site is considered to be a phishing site, else it may be considered legitimate.

$$\text{Rule: IF} \begin{cases} \text{URL Having @ Symbol} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Use of URL shortening service:** This feature is used to check whether a considerably smaller than average URL length still leads to the acquired web page. This check is achieved by using the https redirect on a short domain name.

$$\underline{\textit{Rule}}\text{: IF} \begin{cases} \text{TinyURL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

  The following rule is created to ascertain whether the URL is shorter than the average length. It specifies that if the hostname does not correspond to the length of the URL, the site is a phishing site, else it is legitimate.

- **Some links are pointing to a page:** This feature is used to check the number of links that are pointing to the web page.

$$\text{Rule: IF} \begin{cases} \text{Of Link Pointing to The Webpage} = 0 \rightarrow \text{Phishing} \\ \text{Of Link Pointing to The Web page} > 0 \text{ and} \leq 2 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Use of a non-standard port**: This feature is useful as it can be used to check whether a service such as https is up and running or down. If all the ports are

open, almost any service can run by phishers as they want. As a result, user confidential information is threatened.

$$\text{Rule: IF}\begin{cases}\text{Port \# is of the Preferred Status} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

The following rule is developed to ascertain whether the use of non-standard port that is used for the combination on the Internet. It specifies that if the port does not correspond to the standard port use on the website is a phishing site, else it is legitimate.

Domain identity features:

- **Age of the domain**: This feature is used to extract the information from the WHOIS database and compare it with the information of a suspected phishing site. Most phishing websites are only live for a short period.

$$\text{Rule: IF}\begin{cases}\text{Age Of Domain} \geq 6 \text{ months} \rightarrow \text{Legitimate} \\ \text{Otherwise} \rightarrow \text{Phishing}\end{cases}$$

- **DNS record**: This feature can be used to check the identity of the domain in the WHOIS database records. If the DNS record is not found or is empty, the website is then classified as a phishing web page.

$$\text{Rule: IF}\begin{cases}\text{no DNS Record For The Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Domain registration length**: This feature is used to check how the site is registered. Since phishing websites are live for a short period, this was assuming that trustworthy domains are usually paid for several years in advance.

$$\text{Rule: IF}\begin{cases}\text{Domains Expire on} \leq \text{one} - \text{year} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

The following rule is created to ascertain how long the domain has been registered. It specifies that if the time does not correspond to the original site and the expiration is less than a year, the site is considered a phishing site, else it is legitimate.

- **Sub-domain:** This feature is used to check how the site is registered with a sub-domain. Since phishing websites do not mostly likely create sub-domain, this was assuming that trustworthy domains usually have sub-domain created for some of their services.

$$\text{Rule: IF} \begin{cases} \text{subdomains Expire on} \leq \text{one} - \text{year} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

Source code & Javascript features:

- **Redirect using "//":** This feature is used to check whether "//" exists within the URL path as this indicates that the user will be redirected to another website.

The following rule is created to check whether the URL has an additional at the front of the original URL, such as https://www.ebay.com//https://192.156.100.45.com/. If it does, the site is considered a phishing site. Otherwise, it is a legitimate site.

$$\text{Rule: IF} \begin{cases} \text{the position of the Last Occurrence of "//" in the URL} > 7 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Submits information to email:** This feature is used to check whether a website redirects the user's information to a personal email instead of a server for processing. Phishers usually use this functionality to obtain confidential information by using the mail () or mailto: to trick the user into sending information to the fraudulent email address.

The rule below checks whether the page is rendered to a personal email by the mail () or mailto: functionality on the web page.

$$\text{Rule: IF}\begin{cases}\text{Using "mail ()" or "mailto:" Function to Submit User Information} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **https:** This feature is used to check for the existence of secure communication, whether the security certificate issuer is trusted, and for how long the security certificate has been issued.

Rule:

$$\text{IF}\begin{cases}\text{Use https and Issuer Is Trusted and Age of Certificate} \geq 1\ \text{Years} \rightarrow \text{Legitimate} \\ \text{Using https and Issuer Is Not Trusted} \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing}\end{cases}$$

The following rule is created to ascertain whether there is a secure communication on the and the certificate of the secure communication is less than a year, and the issuer is trusted than the legitimate, else if it uses a secure certificate from an untrusted issuer, then it considered suspicious otherwise the site is a phishing site.

3.2.3 Frame-Based Feature Extraction Approach

Dissolute, online phishing detection using the HTML, source code and URL content is also explored. Three hundred legitimate and phishing website sources were randomly chosen from the Financial Service Authority (FSA) website, APWG and PhishTank website using keyword and the properties of websites using several features, which are outlined below. The legitimate websites are used to discriminate against phishing websites. Principal researchers in the field have used the legitimate 'whitelists' websites (Belabed, Aïmeur and Chikh, 2012; Li *et al.*, 2014; Buber, Ö and Sahingoz, 2017). This process was carried out between 1st October 2016 to 28th February 2017.

- **Iframe redirection**: This feature is used to check the HTML tag used to display additional web pages in the current website. A phisher will take advantage of this feature by making an invisible tag without a frame border.

$$\text{Rule: IF } \begin{cases} \text{Using iframe} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

The following rule is developed to ascertain whether there is conserve information inside a tag used on the website to display information about the web page. It specifies that if the iframe does exit, then the site is a phishing site, else it is legitimate.

- **Disabled right-click**: This feature is used to check whether the right-click function is disabled using the JavaScript so that users cannot save or view the web page's source code. Sometimes the right-click function is also disabled on a fraudulent website that is opened in the menu browser window.

  The rule below is created to check for this function on a web page to determine whether the page is rendered disabled by right-clicking.

$$\text{Rule: IF} \begin{cases} \text{Right Click Disabled} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Use of pop-up window**: This feature is used to check whether users are asked to submit their personal information through a pop-up window, which is unusual to find on a legitimate website. Phishers use this technique for information-gathering to make the request seem more reliable, and they use JavaScript so that the fraudulent pop-ups are reopened if closed until the user fills in the request form.

  The rule below is created to check for this function on a web page to determine whether the page is rendered to a pop-up that requests user information.

$$\text{Rule: IF } \begin{cases} \text{Popup Window Contains Text Fields} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Server form handler (SHF)**: This feature is used to check whether the domain name in the server form handler (SHF) is different from the domain name of the web page. Most e-banking websites usually contain an SHF, and most phishing sites avoid the use of the SHF or refer to a different domain.

$$\text{Rule: IF} \begin{cases} \text{SFH is "about blank" Or Is Empty} \rightarrow \text{Phishing} \\ \text{SFH Refers to A Different Domain} \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

- **Website forwarding**: This feature is used to check how many times a website has a redirect. Generally, a legitimate site does this one time, whereas a phishing site does this more than four times.

  The rule below is created to check the number of times the URL is redirected to another website. It is less than one time; the site is considered legitimate; if the number is higher than two but less than four, the website is judged to be suspicious, else it is considered a phishing site.

$$\text{Rule: IF} \begin{cases} \text{redirect Page} \leq 1 \rightarrow \text{Legitimate} \\ \text{of Redirect Page} \geq 2 \text{ And} < 4 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

- **Link in script & meta**: This feature is used to check that the tag on the website is linked to the same domain of the web page. Phishers can hide URLs by using script and meta to codes to conceal fraudulent sites.

  The following rule is created to check the percentage of links to the actual domain of the website. If the percentage is lower than 17%, the site is considered legitimate if it is higher than 17% but less than 81%, the website is deemed suspicious, and else it is considered a phishing site.

Rule:

IF

$$\begin{cases} \%\text{ of Links in " < Meta > "," < Script > " and " < Link>"} < 17\% \rightarrow \text{Legitimate} \\ \%\text{ of Links in < Meta > "," < Script > " and " < Link>"} \geq 17\% \text{ And} \leq 81\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

- **Layout similarity**: This feature is used to check the percentage of layout similarity of the web page.

  The rule below is developed to check the percentage of layout similarity of the website; if it is less than 31%, the site is considered legitimate, if the

percentage higher than 31%, but less than 67% the website looks suspicious, else it is considered a phishing site.

$$\text{Rule: IF} \begin{cases} \%\ \text{of Layout similarity } < 31\% \ \rightarrow \ Legitimate \\ \%\ \text{of Layout similarity } \geq 31\% \text{ And } \leq 67\% \ \rightarrow \ \text{Suspicious} \\ \text{Otherwise} \rightarrow \ \text{Phishing} \end{cases}$$

- **Style similarity**: This feature is used to check the percentage of style similarity of the web page.

  The rule below is designed to check the percentage of style similarity of the website. If the percentage is lower than 31%, the site is considered legitimate; if it is higher than 31% but less than 67%, the website is judged to be suspicious, else it is considered to be a phishing site.

$$\text{Rule: IF} \begin{cases} \%\ \text{of Style similarity } < 31\% \ \rightarrow \ Legitimate \\ \%\ \text{of Style similarity } \geq 31\% \text{ And } \leq 67\% \ \rightarrow \ \text{Suspicious} \\ \text{Otherwise} \rightarrow \ \text{Phishing} \end{cases}$$

3.2.4 Image Identity Features Extration Approach

An investigation also occurred as to whether the use of visual techniques would make phishing detection more efficient. Such techniques require both image analysis and image matching. Since considered that image matching would be a suitable approach for the detection of new phishing sites, we focused on image matching using the SIFT image matching algorithm for feature extraction. This process was also carried out in parallel with the frame features between the 2nd of November 2016 to 29th February 2017.

- **Favicon**: This feature is used to check the icon associated with a particular web page and to check whether the icon is loaded from a domain other than that shown in the address bar.

$$\text{Rule: IF} \begin{cases} \text{Favicon Loaded From External Domain} \rightarrow \ \text{Phishing} \\ \text{Otherwise} \rightarrow \ \text{Legitimate} \end{cases}$$

The following rule is created to ascertain whether the small image from the address bar of a website is loaded from an external link. It specifies that if the image does not correspond to the site and it is load from another domain, then the website is a phishing site, else it is legitimate.

- **Image size**: This feature is used to check the size of the images on the website.

$$\text{Rule: IF}\begin{cases}\text{The Image Size Is Not Included In URL} \rightarrow \text{Phishing}\\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

- **Alternative text**: This feature is used to check whether a certain percentage of alternative text is used on the website.

The rule below is created to check the percentage of alternative text used on the website. If the percentage is lower than 22%, the site is considered legitimate; if it is higher than 22% but less than 61%, the website is deemed suspicious, else it is considered to be a phishing site.

$$\text{Rule: IF}\begin{cases}\%\text{ of Alternative Text} < 22\% \rightarrow \text{Legitimate}\\ \%\text{ of Alternative Text} \geq 22\% \text{ and } 61\% \rightarrow \text{Suspicious}\\ \text{Otherwise} \rightarrow \text{feature} = \text{Phishing}\end{cases}$$

- **Use of onMouseOver**: This feature is used to check if JavaScript is used to show users a fake URL in the status bar. This function is used to conceal the real identity of a fraudulent URL in the status bar.

$$\text{Rule: IF}\begin{cases}\text{onMouseOver Changes Status Bar} \rightarrow \text{Phishing}\\ \text{It Does't Change Status Bar} \rightarrow \text{Legitimate}\end{cases}$$

The following rule is created to ascertain whether the site display abnormal content using JavaScript. It specifies that if the content change on the status bar and does not correspond to the URL, the site is a phishing site, else it is legitimate.

- **Login form**: This feature is used to check whether there is an obstructive login form on the website.

$$\underline{\textit{Rule}}\text{: IF}\begin{cases}\text{obscreLoginForm} \rightarrow \text{Phishing}\\ \text{Otherwise} \rightarrow \text{Legitimate}\end{cases}$$

The following rule is developed to ascertain whether there is an abnormal form or not on a web page. It specifies that if the website contains obscure login form does not correspond to the site, and then the website is a phishing site. Otherwise, it is legitimate.

All of the above features are used to compare legitimate websites against phishing sites. The features mentioned above are used with our machine learning algorithm to compare the clone website with the legitimate site in order to prevent users from falling victim to fraud when they perform their activities online. Moreover, since the phishing web website is short-lived, the domain identifies features such as the age of the domain, domain registration length, and DNS record are particularly crucial as these enable the system to distinguish a legitimate site quickly from a phishing website.

3.2.5 Features Mining

In this study, a subset of initially selected features is used for training, testing and validating the classifier (Abunadi, Akanbi and Zainal, 2013). The most popular feature selection methods in the literature are the Chi-square (x^2), and information gain (IG).

Chi-Square:

The Chi-Squared test is used to determine whether there is a significant difference between the expected frequencies and the observed frequencies in one or more categories. The Chi-square statistic is the sum of the square of the difference between the experimental data and the data obtained by making a calculation based on a model, where each square is divided by the corresponding data obtained from the model. The Chi-square is used to evaluate features frequency, and this is done by computing the Chi-square statistics of classes to assign the best value for each feature (Gaunt, 2016) by using the following equation:

$$X^2 = \sum \frac{(q_e - q_{e,m})^2}{q_{e,m_i}}, \qquad (\mathbf{3.1})$$

where $q_{e,m}$ is the symmetry capacity obtained by calculating from the model, and q_e is the experimental data of the steadiness in frequency. If the data from the model is similar to the experimental data, X^2 will be a smaller number compare to the original value. If they are different, X^2 will be a more significant number. Therefore, it is necessary also to analyse the dataset using the Chi-square test to confirm the value of the best-fit feature and the frequency at which they occur for the phishing the detection system.

Information Gain:

The information gain (IG) is one of the features ranking metric prominently used for many text features classification techniques that decreases the size of the features by calculating the value of each attribute and ranking them. In other words, IG selects elements through scores (Zeng, Jiang and Neapolitan, 2016). Information gain is one of the most comprehensive approaches to employ as a term importance criterion for text document data. This technique is applied to the feature after the Chi-square has been applied to improve the effectiveness of the features. It is based on information theory, and it expresses in term t in the following equation:

$$IG_{(t)} = -\sum_{i=1}^{|c|} P(c_i) \log P\,(c_i) + P(t) \sum_{i=1}^{|c|} P(c_i|t) \log P\,(c_i|t) +$$

$$P(\bar{t})\, X \sum_{i=1}^{|C|} P(c_i|\bar{t}) \log P(c_i|\bar{t}) \quad \mathbf{(3.2)},$$

where c_i represents the i^{th} category, $P(c_i)$ is the probability of the i^{th} category, $P(t)$ and $P(\bar{t})$ are the probability that the term *t* appears or not in the document, respectively, $P(c_i|t)$ is the conditional probability of the i^{th} grouping given that term *t* appears, and $P(c_i|\bar{t})$ is the conditional probability of the i^{th} grouping given that term *t* does not appear.

In this study, before a numerical value assigns to features, each term within the text is ranked in descending order according to its importance to the classification process by using the IG method. Feature mining aims to identify and examine the attributes that are present in phishing sites. Several features need to be considered by a feature

mining technique in order to improve the overall performance of phishing detection schemes. The unusual characteristics that usually occur in phishing websites include the appearance of some different symbols in the URL and some irregularities in the HTML form and title elements (Zareapoor and Seeja, 2015). Therefore, the extraction of such attributes from these websites will enhance the ability of phishing detection tools (Choo *et al.*, 2016).

The properties of a web page are based on its features, which can be extracted from the source code of the Internet page or removed from the URL (Abunadi, Akanbi and Zainal, 2013). The number of HTML form tags might act as an indicator for identifying a phishing website. The features of a URL can be in the form of multiple tokens that constitute double features, such as some dots, the length of the URL, the existence of an IP address in the URL and a URL with HTTP and an SSL (Mohammad, Thabtah and McCluskey, 2012). The classification of text is an enormous task because there is a large pool of words that needs to be checked, which makes it very hard to classify text quickly and comprehensively. Therefore, text reduction techniques are often used to reduce the size of the text using information gain (IG), or in other words, to transform the text data into a shorter, compact, and predictive format. The two main techniques that were used to reduce the size of text data are Chi-square and information gain.

3.2.6 Feature Comparison

The features are compared with the existing studies in feature-based approaches in terms of diversity and size. Overall, the outline of the features in this study was selected to reduce feature redundancy and use the efficient feature that best for phishing detection. In another instance, Haruta, Asahina and Sasase (2017), only consider two elements of website for phishing detection such as the visual similarity and CSS, while others considered image or JavaScript, whereas our new study consider all possible aspect of website which include the text component, the frame element, and the image

features. Therefore, the technique is used to develop a code that extracts this feature relevant features as the only method to get the source from the website. Also, features were extracted by using a knowledge-based on frame component. A manual download, exploring, identifying and recording was performed during the feature collection process. The strategy was used in order to facilitate accuracy and to reduce the complexity in developing a scheme that will detect phishing efficiently.

3.3 Proposed Design

The proposed approach is a features-based offline model. Features-based models utilise machine learning techniques: for example, one applies a Neuro-fuzzy system that functions similarly to a Mamdani fuzzy model, with three inputs. The Mamdani neuro-fuzzy system itself uses a supervised learning technique (back-propagation) to acquire the parameters of the membership functions (Nguyen, Nguyen and To, 2016). Functionality can be optimised by applying an adaptive neuro-fuzzy inference system (ANFIS) that is functionality similar to the Sugeno fuzzy model with three sets of inputs, which applies hybrid techniques with back-propagation to obtain the membership functions and least-mean-squares estimation to fix the coefficients of the linear mixtures in the rule inferences (Çakıt and Karwowski, 2017).

The proposed approach for website phishing detection was based on using the aforementioned 35 features of the site which are stored in a local Excel table (Adebowale, 2019). The table is a knowledge model database stored offline and newly loaded site features are compared against the features stored in the knowledge model database. If this comparison is unable to detect any similarity, then ANFIS machine learning is used for the decision-making process. An offline version of the image dataset was collected using scale-invariant feature transform (SIFT) image matching to manage the features for image size, noise and illumination; these features were used to identify an object from among many other objects when attempting to locate the image in the testing dataset. These features were extracted and stored as a

dataset, which is used for training and testing. The phishing website data was collected from PhishTank and the WHOIS between 2nd of November 2016 to 29th February 2018.

Other machine learning techniques of relevance here include deep learning: one is to apply the Long short-term memory (LSTM) algorithm in the part of the structure of the scheme that takes the input from a URL as a character sequence and predicts whether the link is a phishing or legitimate website. In the Convolutional Neural Network (CNN) concept, the weights are shared in a temporal dimension, which leads to a decrease in computation time. The general matrix multiplication in the standard neural network (NN) is therefore replaced in the CNN. The images that are extracted from legitimate and phishing websites were collected from 10th August 2018 to 30th December 2018 numbered well over 10,000.

3.3.1 Inputs Features

The current work identified 352 main phishing website elements that can assist in distinguishing between legitimate and phishing sites. These features were used on the off-line modelling applying ANFIS which prunes the number down to 35 because of their uniqueness to phishing detection complimentary to the removal of redundant features, markedly improving the efficiency of the proposed scheme.

3.3.2 Offline Intelligent Phishing Detection System Based on FIS

The intelligent phishing detection system (IPDS) offline approach will have four essential components (Fig. 3-2) for learning and reasoning, which include; fuzzification, rule-base, inference engine and defuzzification. Fuzzy inference systems can employ human expertise by loading this essential component as rules and performing fuzzy reasoning to deduce the overall output value. There two types of fuzzy inference system scheme: the Mamdani model and the Sugeno model (Karaboga and Kaya, 2016). Initially, Mamdani's fuzzy inference method was regarded

as the standard the fuzzy methodology, and it was the first control system built using fuzzy set theory.

This process of generating membership value for the fuzzy variable using membership functions begins by taking the crisp inputs from the 35 characteristics and factors which indicate the fake phishing website and determining the level at which these inputs belong to each appropriate fuzzy set. The crisp input is always a numeric value limited to the universe of discourse, whilst the fuzzy detection model assigns degrees of membership to the decision-making process for identifying phishing websites.

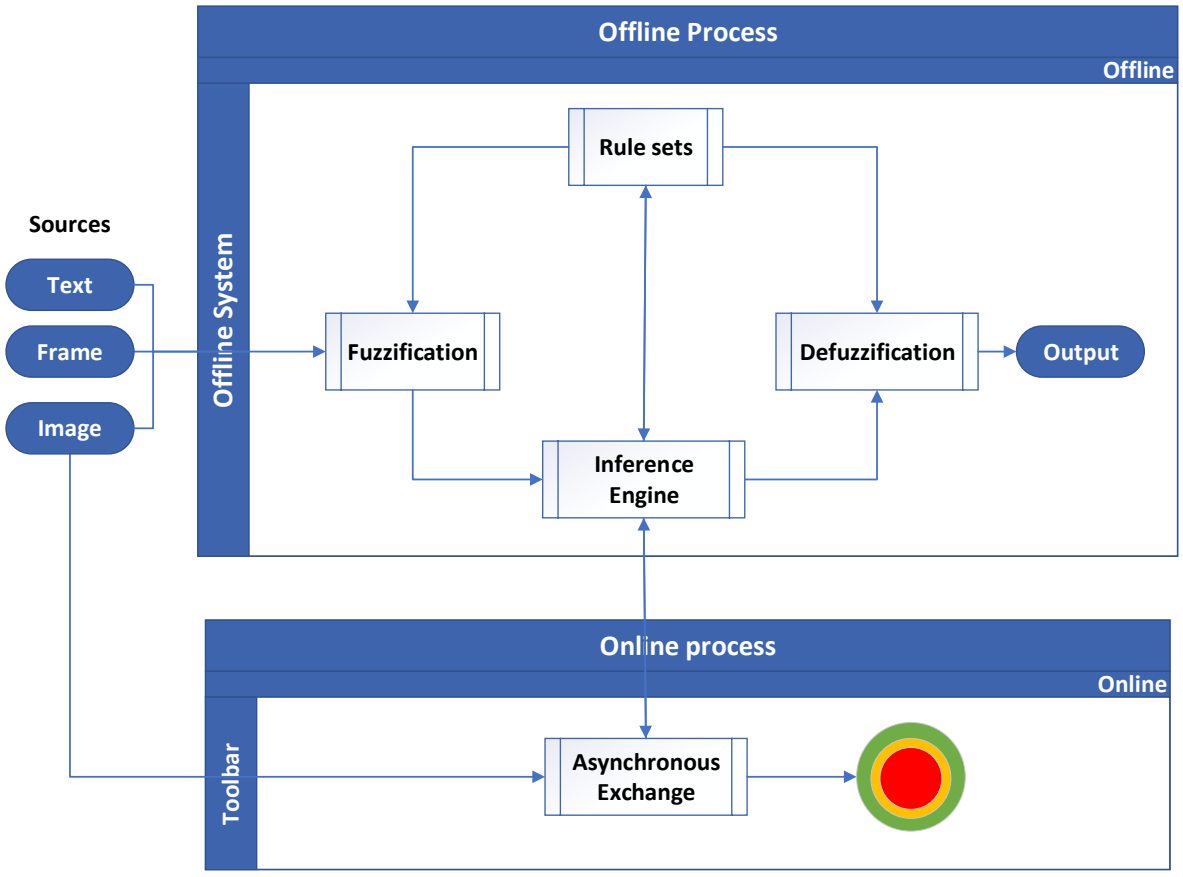


Fig. 3-2: Intelligent Phishing Detection System Architecture compose off-line and online Structure

The essential advantage of using fuzzy logic techniques is that they can use the linguistic variable to represent major phishing characteristics or indicators and their relations with possible phishing web pages. Descriptors which come in the form of high, medium and low qualifiers are assigned to a variety of values for each primary phishing feature indicator. The descriptor forms the basis for acquiring expert input, which is based on the vital phishing element that indicates the existence of a phishing website. The range assigned to the input is considered and divided into fuzzy sets. For example, the value of the linguistic variable for the pop-up window ranges from high and medium to low.

The level of fitting of the values of the variables to any selected class determines the level of membership. A membership function is built for each phishing feature indicator, and this defines how the curve at each point in the input space is mapped to a membership value between 10 and 100. As mentioned above, various values are assigned to phishing indicators, ranging from high to medium to low. On the other hand, the values for websites range from phishing to suspicious to legitimate. Also, the input values range from 10 to 100, while the output values range from 0 to 100.

The linguistic indicator is used to signify one of the critical phishing element indicators, the pop-up window. The plot of the fuzzy membership function for the pop-up window is shown in Fig. 3-3 to 3-6 below. The x-axis in the plot illustrates the range of possible values for the corresponding critical phishing feature indicators of high, medium and low. The y-axis illustrates the level to which a value for the leading phishing element indicator is shown by the linguistic descriptor. The plot of the membership function of the pop-up window at a distance of 54 cm is considered low with a membership of 35%, but it is also considered to be medium with a membership of 65%. In other words, the distance of 54 cm for the pop-up window is considered both low and medium to varying levels of certainty, which is a distinctive aspect of fuzzy logic, in contrast to binary logic that artificially imposes black and white limitations on any interpretation. The fuzzy representation is possible matches human cognition,

which means that facilitating expert input and more reliable in place of experts' understanding of the underlying dynamics (Aburrous *et al.*, 2010).

The range for this fuzzy variable is specified depending on the level of risk associated with the phishing feature. A legitimate website cannot allow too many pop-up windows that ask for vital information that can be used for phishing purposes. The unwanted pop-up is the reason there is a fuzzy set range that protects against such occurrences and phishing attacks with fuzzy values ranging from high to medium and low as appropriate.

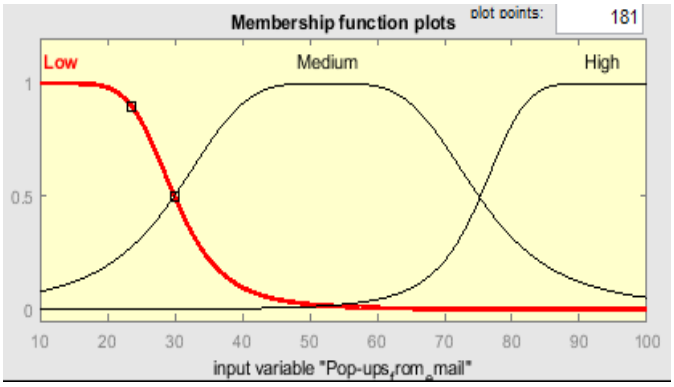


Fig. 3-3: Input Variable for Pop-up Window Component

Table 3-1: Value Range for Pop-up Window

Linguistic value	Numeric range
Low	[0,24,30,54]
Medium	[30,40,65,75]
High	[53,75,80,100]

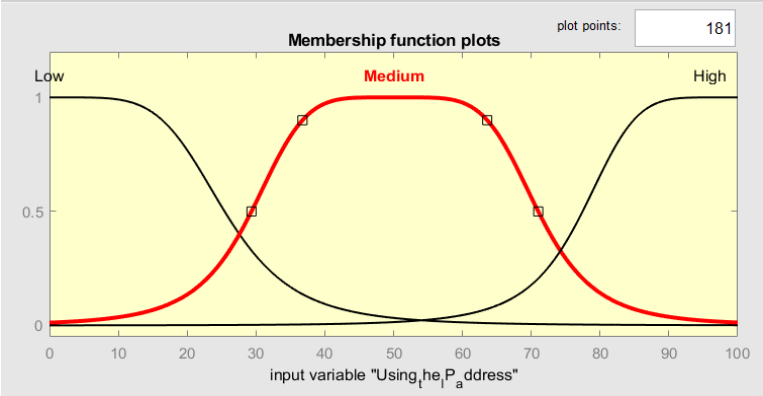


Fig. 3-4: Input Variable for Using the IP Address

Table 3-2: Value Range for Using the IP Address

Linguistic value	Numeric range
Low	[25.44 2.5 -1.39e-15]
Medium	[20.83 2.5 50.18]
High	[20.8 2.5 98.26]

The plot of the membership function of the using IP address at a distance of 50.18cm (Fig. 3-4) is considered medium with a membership of 40%, but it is also considered to be high with a membership of 60%. In other words, the distance of 50.18cm for the using IP address is considered both medium and high to varying levels of certainty (Table 3-2).

The plot of the membership function (Fig. 3-5) of the URL has “@” symbol at 94cm is considered high (see Table 3-3) with a membership of 60%, but it is also considered to be high with a membership of 40%. In other words, the distance of 94cm for the URL has “@” symbol is considered both high and medium to varying levels of certainty.

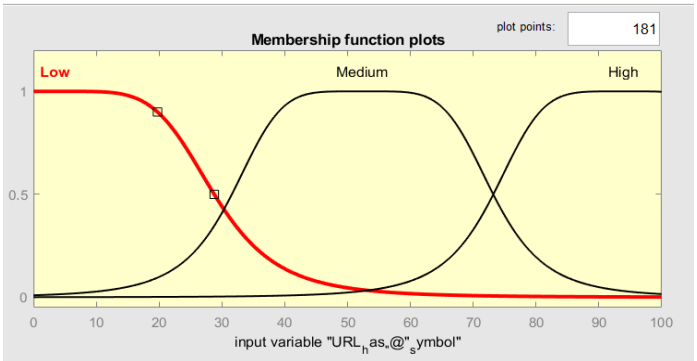


Fig. 3-5: Input Variable for URL has @ Symbol

Table 3-3: Value Range for URL has @ Symbol

Linguistic value	Numeric range
Low	[25.4 2.5 3.38]
Medium	[20.8 2.5 52.38]
High	[20.8 2.5 94]

The plot of the membership function (Fig. 3-6) of the alternative text at 90.6cm is considered high (see Table 3-4) with a membership of 90%, but it is also considered to be high with a membership of 10%. In other words, the distance of 90.6cm for the alternative text is considered both high and low to varying levels of certainty

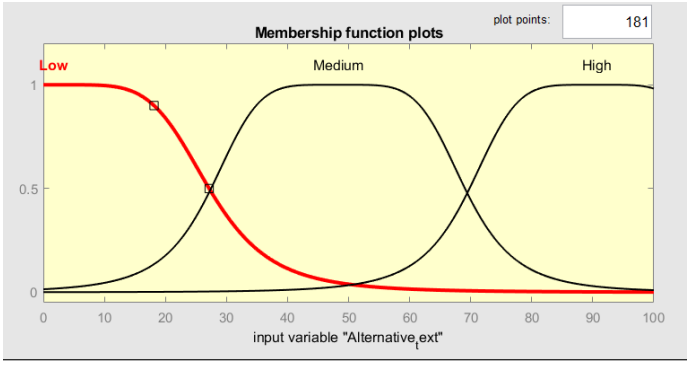


Fig. 3-6: Input Variable for Alternative text

Table 3-4: Value Range for Alternative Text

Linguistic value	Numeric range
Low	[25.4 2.5 1.76]
Medium	[20.8 2.5 48.27]
High	[20.8 2.5 90.6]

The same techniques are used for all the other essential phishing website feature indicators. The value that is assigned to the range of a fuzzy variable is derived and tuned from a series of phishing experiments done.

3.3.3 Optimising Offline IPDS Base on ANFIS

In order to optimise the feature-based offline model of this work cross-validation methods/experiments were undertaken, which is later present in chapter 4. However, a 5-fold cross-validation was optimal result which involved seven major experimental runs where five combinations of parameters were used to tune for a better solution. The ANFIS-based method involves the use of a network structure (see Fig. 3-2) that facilitates the systematic computation of gradient vectors, where it combines the least-squares and the gradient descent methods by utilising a useful hybrid learning technique to derive the output error and optimise the detection of the phishing website (Çakıt and Karwowski, 2017). As mentioned earlier, the ANFIS is a model that accept features as an inputs selection and it trains the data using a least-squares application (Hosoz, Ertunc and Bulgurcu, 2011). The model that is most frequently used with the ANFIS is the Sugeno model and so is employed in this current research; it has differentiable functions that can learn the fuzzy inference primary system from data and it can be easily understood (Hosoz, Ertunc and Bulgurcu, 2011). The detailed features of each layer of the zero-order Sugeno fuzzy inference system with inputs x and y and two rules are as follows:

IF x_1 $is\ A_1$

AND $x_2\ is\ A_2$

…….

AND $x_m\ is\ A_m$

THEN $y = f(x_1, x_2, \ldots\ldots x_m)$

where $x_1, x_2, \ldots\ldots x_m$ are input variables; $A_1, A_2, \ldots\ldots A_m$ are fuzzy sets, and y is either a constant or a linear function of the input variables. However, if y is constant, the zero-order Sugeno fuzzy model, which the resulting of rule-based is specified, could be obtained by a singleton (Barraclough *et al*., 2013). In the study, the input of the fuzzy inference system is applied to the antecedence of the fuzzy rules (Table 3-5). Meanwhile, the fuzzy rule has multiple behaviours; the operator (AND or OR) is used to acquire a single number that signifies the result of the evaluation of the property of each phishing website feature indicator. The AND fuzzy operation intersection is used to determine the conjunction between the rule's behaviour.

After identifying the risk associated with the phishing website and the important phishing features indicators, the next step is to identify how the phishing website probability varies as a functioning part of the phishing indicator. The fuzzy rules were developed in the form of ***if…then*** statements that relate to the varying level of possibility of a website being a phishing website based on an assessment of the leading phishing element indicators. Various resources were used to evaluate the features and factors of phishing websites and the associations and relationships between them to develop the scheme, such as an analysis of anti-phishing tools, website experiments, a detailed questionnaire, and web surveys. These available resources helped the author to become an 'expert' in building phishing website fuzzy rules. The rules are unified, and the membership functions combined into a single fuzzy sets output using ANFIS.

The ANFIS hybrid learning algorithm is used to tune the parameters of a Sugeno-type fuzzy inference system using a combination of the least-squares and back-propagation gradient descent methods to model a training dataset. Hybrid features are

used because they can represent phishing attack techniques and strategies. These features are used as training and testing input data for the neuro-fuzzy inference system so that it can generate the fuzzy ***if….then*** rules (Table 3-5) to differentiate between legitimate, suspicious and phishing websites.

Table 3-5: Sample of Rules Used in Intelligent Phishing Detection

Rule#	(Component 1) Using the IP address	(Component 2) Long URL	(Component 3) Short service	(Component 4) Using the @ sign	(Component 5) Using the double splash	Text features phishing risk
1	Low	Low	Low	Low	Low	Legitimate
2	Moderate	Moderate	Moderate	Moderate	Moderate	Suspicious
3	High	High	High	High	High	Phishing
4	Low	Low	High	High	Low	Suspicious
5	High	Low	Low	High	High	Phishing
6	Low	Low	Moderate	Moderate	Moderate	Suspicious
7	Low	Moderate	High	Moderate	High	Phishing
8	Moderate	Low	Low	Moderate	Moderate	Suspicious
9	High	Moderate	High	Low	Moderate	Phishing
10	Low	Low	Low	Moderate	Low	Legitimate
11	Low	Low	Moderate	Low	Moderate	Legitimate
12	Low	High	Low	High	Low	Suspicious
13	Moderate	Low	High	Moderate	High	Phishing
14	Low	Low	Low	Moderate	High	Legitimate
15	Low	High	High	Low	High	Phishing

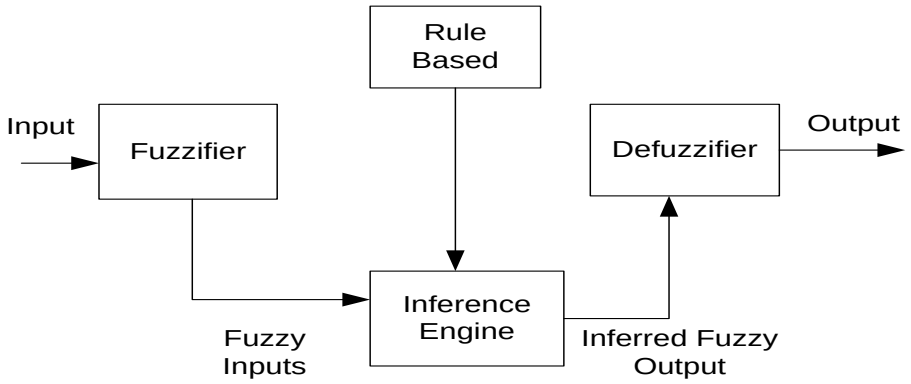


Fig. 3-7: Block Diagram of Intelligent Phishing Fuzzy Inference System Structure

The structure of the intelligent fuzzy inference system has five function layers (Fig. 3-7) that are used in the decision-making process as follows:

Layer 1: Input Layer

Each node in the input layer is assigned a parameter, which includes three membership functions. The neurons in this layer quickly transmit visible crisp indications straight to the next tier (GüNeri, Ertay and YüCel, 2011). Equation (1) below shows how this is done, where a_i, b_i, c_i (*i = 1, 2, …. nth*) is the parameter set, $\mu_A(x)$ is the membership function of fuzzy set A_i and X is the input. As the value of the parameters in the set change, the shape of the bell-shaped function varies (Fig. 3-8), so these types of a parameter is referred to as the essential parameters (GüNeri, Ertay and YüCel, 2011).

$$\mu_A(x) = \frac{1}{1+|\frac{x-c_i}{a_i}|^{2b_i}} \qquad (3.3)$$

Layer 2: Fuzzification Layer

A node in the fuzzification layer acts as a membership function to represent the terms of the respective linguistic label, such as phishing, suspicious and legitimate, and it assigns a value for each key phishing feature indicator (Dariane and Azimi, 2016). The valid range of the inputs (*X, Y*) is considered and is divided into the fuzzy set. The output value of the input layer is fed into

the fuzzification layer, and Gaussian membership functions are used with two parameters: variance and mean. Also, $\mu_{A_i}\ and\ \mu_{B_i}$ is the category of the fuzzy sets. The output function of this node is the product to which the input belongs and the given membership function. Equation 3.4 shows how the fuzzification of the inference system is determined.

$$O_i^2 = w_i = \mu_{A_i}(x) \cdot \mu_{B_i}(y), \quad i = 1,2 \tag{3.4}$$

Where the values A and B are the input parameters. Every node in this layer is a fixed node whose output is the product of all the incoming signals. Each node's output represents the firing strength of a rule (Hosoz, Ertunc and Bulgurcu, 2011). The output is used to determine the number of rules in the next layer.

Layer 3: Rule Generation Layer

Every node in the rule generation layer is a fixed node labelled *N*. The i^{th} of this node is to calculate the ratio of the i^{th}rule firing strength to the sum of all the rules' firing strengths. This is done by a rule-based layer that consists of ***if…then*** statements that are related to possible phishing sites at different levels that get an input w_i from the individual fuzzification i^{th}nodes and calculate the strength of the rule they represent (Karaboga and Kaya, 2016), as shown in Equation 3.5 below. The output of this layer is called the normalised firing strength.

$$O_i^3 = \overline{w_i} = \frac{w_i}{w_1+w_2}, \ i = 1,2, \ w_i = \text{input} \tag{3.5}$$

A sample of rule-based functionality is presented in Table 3-5.

Layer 4: Normalisation Layer

This layer is where normalisation occurs (Fig. 3-7). All the neurons in this layer are connected to an individual normalisation neuron, as illustrated in Fig.3-7.

The output neuron from the rule-based layer is fed into this layer, and the normalisation of the neuron's firing strength is resolved. The power of the normalised firing neuron is the percentage of the firing strength as instructed and the sum of the firing force of every rule (Barraclough *et al*., 2013), as illustrated by Equation 3.6:

$$O_i^4 = \overline{w_i} f_i = \overline{w_i}(p_i x + q_i y + r_i), \qquad (3.6)$$

where $\overline{w_i}$ is the normalised firing strength from layer 3 and $\{p_i, q_i, r_i\}$ are the parameters settings, which are referred to as essential parameters (Hosoz, Ertunc and Bulgurcu, 2011).

Layer 5: Defuzzification Layer

The fifth and final layer is the defuzzification layer (Fig. 3-7). Here the neuron combines the sum of all the output neurons and produces the ANFIS output, as shown in Fig. 3-8. The single node in this layer calculates the total output as the summation of the contribution from each rule. The input for the defuzzification process is the aggregate output of a fuzzy set, and a result is a number. The output is the phishing website risk category, which is defined in the fuzzy set as phishing, suspicious, or legitimate. The fuzzy output set is then defuzzified to arrive at a scalar value, as shown in Equation 3.7:

$$O_1^5 = Overall\ output = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (3.7)$$

in which the three risk categories are defined as follows:

- ***Legitimate:*** The highest guarantee that the website is a legal, genuine web page, and there no reason to think otherwise so that a user can conduct his/her activities online safely on this website.

- ***Suspicious:*** There is reasonable doubt about the authenticity of the website, and some level of caution should be exercised when dealing with this site because it could be risky using it.
- ***Phishing:*** This is the highest guarantee that the website is a fake phishing website that will hack all the user's confidential information when he/she is using the website to conduct his/her online activities.

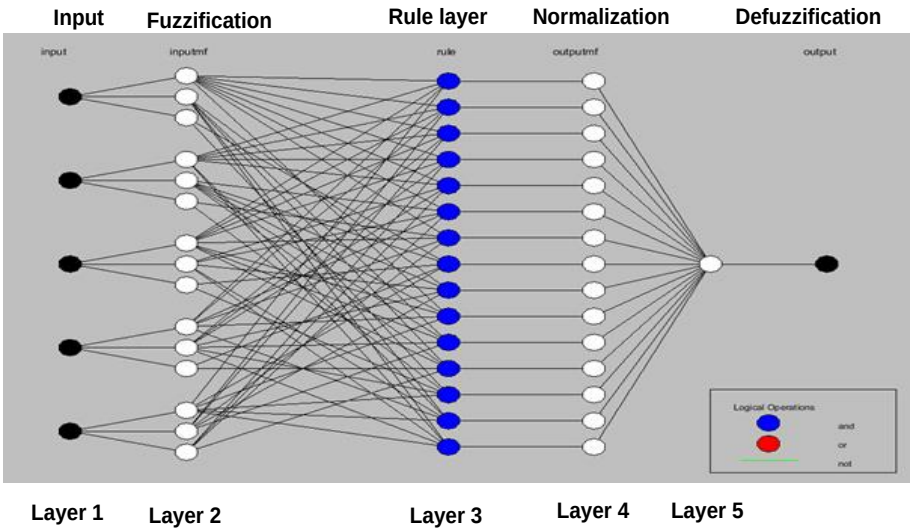


Fig. 3-8: Intelligent Phishing Detection Fuzzy Inference System Structure

3.4 Evaluating IPDS Modelling and System Detection

In this work, it was determined that a reasonable phishing detection rate could be achieved based on seven criteria (Shahriar and Zulkernine, 2012): the search index, URL content, web address bar, image identity, domain identity, source code & JavaScript and page style & layout identity. The selection of phishing features requires careful deliberation. Table 3-6 shows that there are a different number of components for each criterion. The grouping process was undertaken to simplify the fuzzy model since dealing with 35 website phishing features as a whole can make the fuzzy rule evaluate very complicated and time-consuming. This was the group and categorised into 35 phishing website features and factors various criteria. The search index has

four elements, the security and encryption have eight, the URL content has six parts, while there are five elements for the web address bar and image identity, four elements for domain identity, three features for source code & JavaScript, and eight for page style & layout identity. Therefore, there are 35 critical components in total. These elements were selected as the best for the detection of phishing and to improve the time of discovery too.

A laying process was also implemented in these phishing websites to enhance and improve the result of the phishing website detection output. As illustrated in Table 3-6, the proposed intelligent phishing detection scheme has three layers. The first layer contains only the text identity component with the search index criterion with a weight equal to 0.3, while the URL content criterion is assigned a weight equal to 0.3 due to its importance; a user could follow this link to a vulnerable site. The security and encryption are assigned a weight of 0.2. The web address bar, image identity, domain content, page style & layout identity and source code & JavaScript each have a weight equal to 0.1. A weight assigned to those features according to their influence and frequency using the Chi-Square and some literature.

The seven criteria were prioritised according to their importance by using weights to rate each criterion. The rankings and weights were determined from a case study, website phishing experiments, an analysis of anti-phishing tools, a web survey, phishing quizzes, phishing expert feedback, and the results of a detailed questionnaire. Different parameter values were used in order to identify the most efficient detection approach. The parameter values that were found to provide the best result in our model were as follows, where:

Crisp text is represented as:

d_1= URL content

d_2= Search index

d_3= Security & encryption

d_4 = Domain identity

d_5 = Source code & JavaScript

The crisp frame is represented as:

g_1= Page style & layout identity

The crisp image is represented as:

h_1= Image identity

The intelligent phishing detection rating Z_1 is calculated from the weight parameters as follows:

$$z_1 = (0.3 * d_1) + (0.2 * d_2) + (0.1 * d_3) + (0.1 * d_4) + (0.1 * d_5) + (0.1 * g_1) + (0.1 * h_1) \quad (3.8)$$

However, it should be noted that when selecting the best phishing features, it is also essential to take into consideration that phishing strategies and techniques change with time. Thus, the number of features that could be used for modelling a phishing detection system can vary over time. The architecture of the fuzzy logic inference-based phishing detection model is shown in Fig. 3-8. As shown in the structure figure, the final output website phishing result for this fuzzy model relies on evaluating the fuzzy outputs of these three layers and then combining those results.

There are some challenges attached to phishing websites post-classification. The most challenging concern is the use of the phishing website material and date as a form of information, which has the net effect of increasing the false-negative rate. The age of the dataset is the most substantial problem, primarily regarding the phishing quantity. This is because some phishing websites are short-lived, sometimes lasting only 48 hours (Barraclough *et al*., 2013).

Table 3-6: Components and Layers of Phishing Website Criteria

Criteria	No.	Component	Layer No.
Search index (weight = 0.3)	1	Page ranking	Text layer Sub-weight = 0.3
	2	Google index	
	3	Website traffic	
	4	Statistical report	
Security & encryption (weight = 0.2)	1	Long URL	Text layer Sub-weight = 0.2
	2	Using the IP address	
	3	Abnormal URL	
	4	Abnormal request	
	5	Abnormal anchor	
	6	Existence of "https" in URL	
Web address bar (weight = 0.1)	1	Adding prefix or suffix	Text layer Sub-weight = 0.1
	2	URL has "@" symbol	
	3	Using URL shortening services	
	4	Some links are pointing to a page	
	5	Use of a non-standard port	
Domain identity (weight = 0.1)	1	Age of the domain	Text layer Sub-weight = 0.1
	2	DNS record	
	3	Domain registration length	
	4	Sub-domain	
Source code & JavaScript (weight = 0.1)	1	Redirect using "//"	Text layer Sub-weight = 0.1
	2	Submitting information to an email	
	3	https	
Page style & layout similarity (weight = 0.1)	1	Iframe redirection	Frame Features Approach Sub-weight = 0.1
	2	Disabling right-click	
	3	Using a pop-up window	
	4	Server form handler (SHF)	
	5	Website forwarding	
	6	The link in script & meta	
	7	Layout similarity	
	8	Style similarity	
Image features (weight = 0.1)	1	Favicon	Image Features Approach Sub-weight = 0.1
	2	Image size	
	3	Alternative text	
	4	Mouse over	
	5	Login form	
Total weight			1

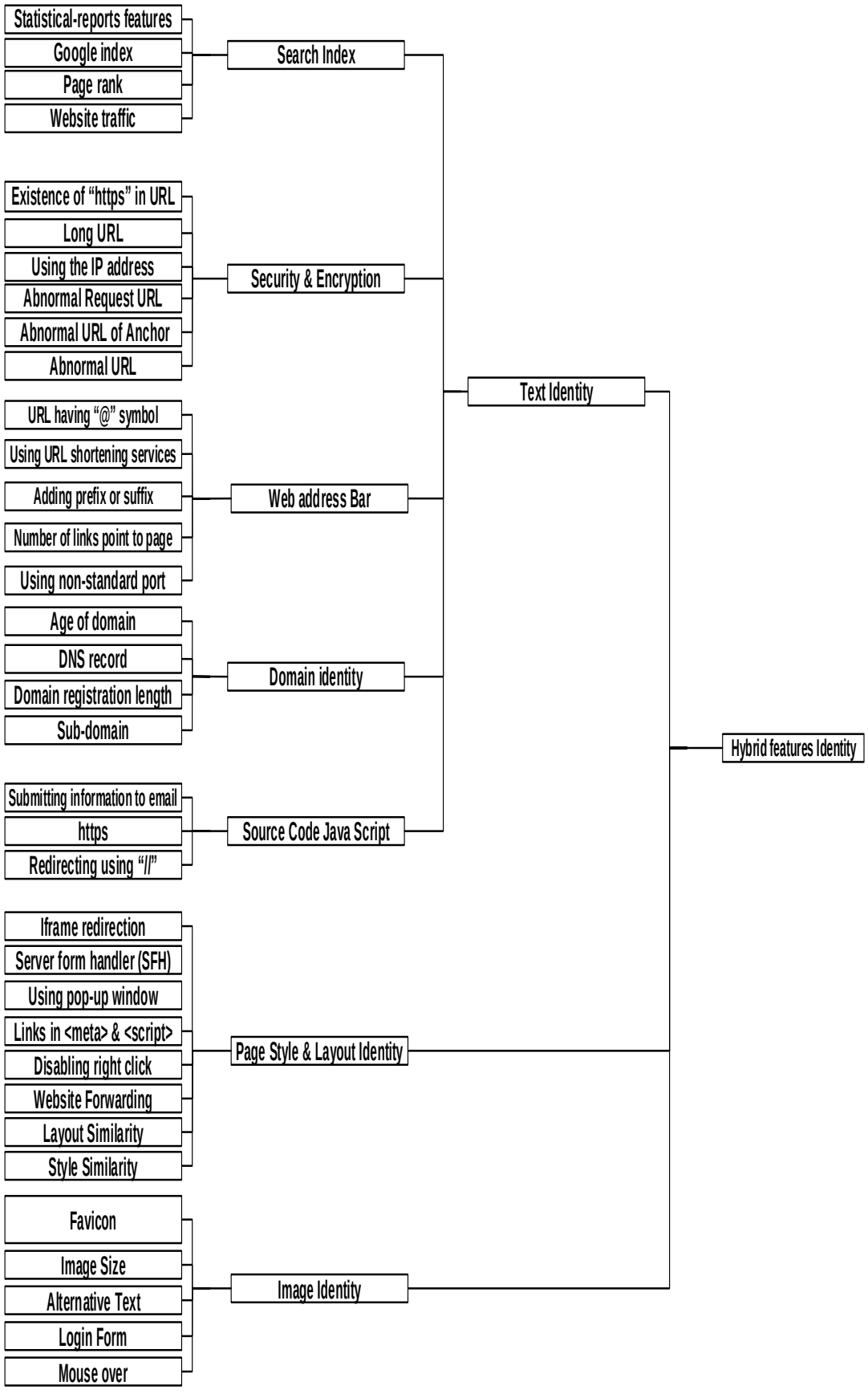


Fig. 3-9: Architecture of ANFIS Model for Evaluation of Phishing Websites

The above approaches are implemented using MATLAB Version: 9.5 fuzzy logic toolbox, which runs on Windows 10 64bit computer with Intel Pentium 3.2GHz and 4GB memory. Features were extracted and normalised to a standard value ranges from 0 to 1 on the y-axis and standard range of 10 to 100 on the x-axis in order for the features to be transformed into features suitable for MATLAB. The MATLAB Version: 9.5 was used because it meets the proposed set objectives.

3.5 Offline Intelligent Phishing Detection System based on Deep learning

A deep learning (DL) algorithm is categorised as a type of unsupervised machine learning algorithm that learns from the data on its own and designs a scheme for future use. This type of algorithm has a high probability of detecting newly generated phishing URLs and does not need manual feature engineering. In recent times artificial intelligent technology has come to power many aspects of modern society, ranging from social networking and web searching to content filtering and e-commerce websites. It also has a presence in consumer products such as cameras and smartphones (Sundermeyer, Schlüter and Ney, 2012).

In the current approach, a deep learning algorithm was used in a phishing website detection system that was based on long short-term memory (LSTM) and the convolutional neural network (CNN). In this system, CNN and LSTM were combined to detect a variety of website elements in order to attempt to identify phishing websites more accurately. Long short-term memory was used to detect extracted features such as the text and frame content of the web page, while the CNN was used to analyse the image features of the website (Xu, Li and Deng, 2015).

3.5.1 Deep Learning CNN+LSTM Structure

LSTM is an adaptive recurrent neural network (RNN), where each neuron is exchanged by a memory cell which is additional to the conservative the neuron on

behalf of an internal structure. It also uses multiplicative units as gates to control the flow of information. The central components of the LSTM architecture are the memory cell, which can maintain its state over some time, and non-linear gate units which regulate the information input and output flow of the network (Greff *et al.,* 2017). Based on the insights derived from secure networks, it is considered that because the LSTM neuron consists of internal cells and gate units, one should not only look at the output of the neuron but also at the internal structure to design original features for LSTM so that it can address classification problems (Hakkani-Tür *et al*., 2016).

Figure 3-9 shows the architecture of an LSTM in which there are three bidirectional LSTM layers, two feed-forward layers, and a Softmax layer that gives the predictions. This fully connected architecture allows us to take advantage of the inherent correlations among connections. Before the second layer in the network, co-occurrence exploration is applied to the connection to learn from the input features. Lastly, back-propagation is applied to the LSTM layer to allow more effective learning (Zhu *et al*., 2016).

Input: The input layer is the entry point for the architecture, which is fully connected to the LSTM layers. The LSTM is its memory cell, which eventually acts as an accumulator of the cell if the input gate is activated. Also, the prior cell status could be forgotten in this process if the forget gate is on. However, the latest cell output will be propagated to the final state will be further controlled by the output gate. The advantage of using the memory cell and gate is to control the flow of information in the gradient will trapped in the cell and be prevented from vanishing too quickly, which is a critical issue in the RNN model (Zhu *et al.*, 2016).

Softmax layer: This function calculates the probability distribution of the k output classes. The layer uses the softmax function to predict the class in which the input features belongs (legitimate, suspicious or phishing). The softmax function ensures

that the network outputs are all between 0 and 1 and that the sum equals one at every timestamp, as shown in Equation 3.9 below where x is the net input.

$$v_j = \frac{f^{x_j}}{\sum_1^k f^{x_k}} \quad for\ j = 1, \ldots . k \qquad (3.9)$$

The features of the phishing website URLs were collected from PhishTank and those of legitimate sites were obtained from Common Crawl. The data consisted of various types of information, but we only extracted the URL and the target site information for the categorisation (Table 3-7). Then the data was tokenised to separate each URL into a series of separate words, all of which were set in lowercase. The tokenised data was then encoded to make it available for training, where the maximum length was set to 75 according to the format, we followed in the later section.

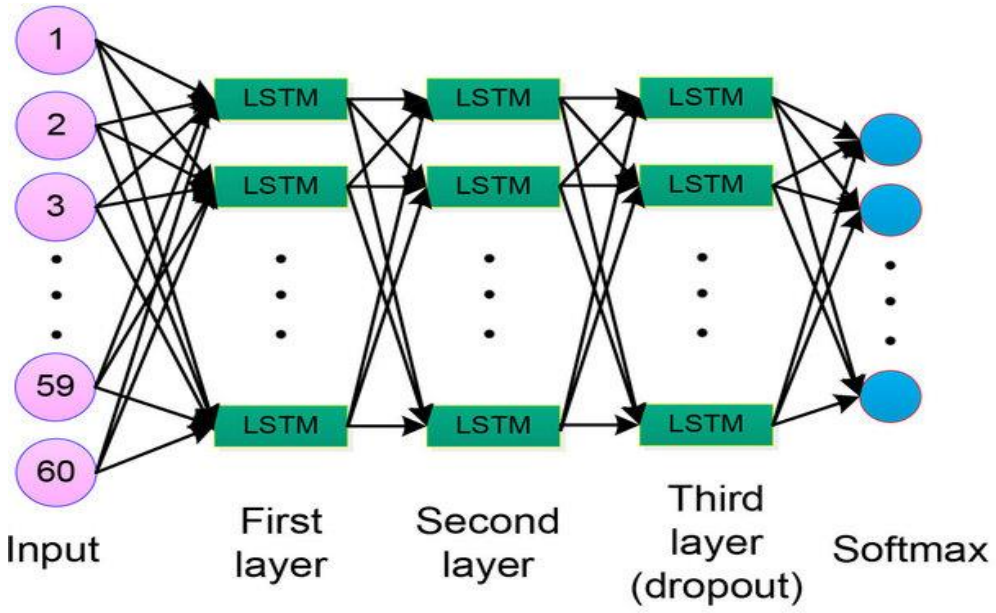


Fig. 3-10: Schematic Structure of Fully Connected LSTM (Source: Zhu et al., 2016)

Table 3-7: The URLs Used in the LSTM

URLs	Category
http://bcpzornaseguras.com/bbvacontinentalpe-enlinea-20938209d23kjdh23d90238d23jwxj23/	Phishing
https://www.google.com/	Legitimate
https://www.microsoft.com/en-gb/	Legitimate
http://kelberdesigner.com/adesao/eng/2.html	Phishing
http://www.stopagingnews.com/wp-admin/js/wells/ibrowellsup/identity.php	Phishing
http://orientality.ro/RENNE/ourtime.com/ourtime.com/ourtime.html	Suspicious
http://bcpzornaseguras.com/	Suspicious
http://www.vinaros.org/locale/es/fb/	Phishing
http://www.vinaros.es/locale/es/fb/	Phishing
http://bcpzonasegura.viai1bcp.com/bcp/0peracionesnlinea/	Phishing
http://riquichichichi.tk/ptm/web/	Phishing
http://unitedstatesreferral.com/santos/gucci2014/gdocs/gucci.php?Acirc=A?A?=A?Auffe0=	Phishing
http://www.Legitimategovbr.com/SIIBC/	Phishing
http://201.73.146.167/teste/	Phishing
https://my.anglia.ac.uk/CookieAuth.dll?GetLogon?curl=Z2F&reason=0&formdir=3	Legitimate
https://uk.yahoo.com/?p=us	Legitimate

There are three main components in the learning process of a CNN: equal representation, sparse interaction and parameter sharing (Goodfellow, Bengio and Courville, 2016). The CNN is different from the standard NN, which draws out the connection among the input and output units from matrix development.

In contrast, CNN decreases the computational load with a thin interface where the kernels are made slighter than the inputs and are used for the entire image. Also, in the CNN, the idea behind parameter allocation is that, rather than learning a detached set of parameters at each location, the CNN only needs to learn a set of features, which allows the CNN to perform better than the NN. Also, CNN has a beneficial property called equivariance, which works with parameter distribution so that every time the input changes the output follows suit. Hence, the CNN requires fewer parameters than legacy NN algorithms. This requirement leads to a reduction in memory usage and improves efficiency.

The components of the standard CNN layers are illustrated in Fig. 3-10. The figure shows how the input image is convolved with trainable filters with possible offsets to produce feature maps in the first c-layer. The filters contain a layer of connection weights. In a real sense, there are four pixels in the feature map in a group. These pixels pass through a sigmoid function to produce additional feature maps in the first layer. This process is continued to obtain the feature maps in the following c-layers and convolved layers. Then, at the end of this process, the values of these pixels are rasterised and displayed in a single vector as the input of the network (Arel, Rose and Karnowski, 2010).

The layer that is responsible for the gathering of features when the input of each neuron is connected from the previous layer is called the c-layer. After the local features are extracted, the positional association can then be identified. Also, the function kernel has a slight influence over the activation function that is used by the sigmoid function to achieve scale invariance. Hence, the model uses the filter to connect the series of overlapping available fields and converts the 2D image set from the input to a single element in the output.

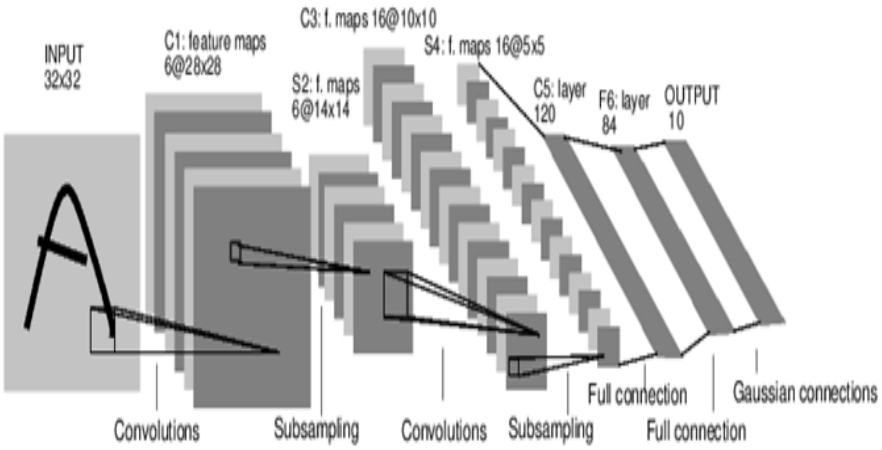


Fig. 3-11: Intelligent Phishing Detection Convolutional Neural Network (CNN) System Structure (Source: LeNet (2012))

However, when overfitting occurs, a pooling process called sub-sampling is utilised to decrease the total size of the signal. This solution has been used for data size reduction in audio compression (Mathieu, Henaff and LeCun, 2013).

The three layers of the CNN architecture are described in more detail below:

Convolution Layer:

This layer contains filter kernels which slide across the image. The kernel is the matrix to be convolved with the input image, and stride length controls how much the filter convolves the input image. This layer performs the convolution of the input image with the kernel by using Equation 3.10. The output of the convolution is called the feature map.

The convolution operation is as below:

$$\boldsymbol{j_k} = \sum_{n=0}^{N-1} \boldsymbol{s_n e_{k-n}} \tag{3.10}$$

where $\boldsymbol{s}$ is an image, e is the filter, and N is the number of elements in $\boldsymbol{s}$. The output vector is j. The subscripts represent the nth element of the vector.

Pooling Layer:

This layer is also called the sub-sampling (Fig 3-10) layer. In order to prevent overfitting, the pooling operation is used to reduce the dimension of the output neurons from the convolution layer and thus reduce computational intensity. This study used max-pooling operation. The max-pooling operation picks only the best value in each feature map in order to reduce the number of output neurons.

Fully Connected Layer:

The previous activation layer is fully connected to this layer. The activation function is always employed after the convolutional layer. The activation function is the operation that maps the output to a set of inputs.

There two types of activation function:

1. **Softmax**: This function calculates the probability distribution of the k output classes by using Equation 3.11. The fully connected layer uses the softmax function to predict the class to which class the input image belongs (legitimate, suspicious or phishing).

$$v_j = \frac{f^{x_j}}{\sum_1^k f^{x_k}} \ for \ j = 1, \ldots . \ k \tag{3.11}$$

where x is the net input value. Note that the output values of v are between 0 and 1, and their sum is equal to 1.

2. **Rectified linear activation unit:** The rectified linear function is an established activation unit for deep learning (Saif, El-Gokhy and Sallam, 2018). This type of activation function is used to apply non-linearity to the network structure.

The Convolutional Neural Network (CNN) approach reduces the weights, thereby decreasing the complexity of the network. Consequently, the feature extraction procedure in a standard learning algorithm can be enhanced by directly importing images into the network as raw inputs. The use of this type of model for the training of the architecture layers led to the success of the first DL algorithms.

Below in Fig 3-11 are some examples of images that were used to train the CNN in the current work.

Fig. 3-12: Sample Images Used for the Training of CNN

3.6 Online Plugin Approach

This section describes the approach taken to building a useful features-based on-line toolbar for phishing detection. An off-line detection model was develop using ANFIS with voice generation user warning interface algorithm incorporated with a clear text directive and colours status with 35 comprehensive features are utilised to alert users. The 35 comprehensive features illustrate in Table 3-6, was used to develop a knowledge model that runs at the background as a toolbar comparing all the requested website against the 35 features to check whether the requested website is legitimate, suspicious or phishing. The feature-based online model has three essential parts, including extractor algorithm, knowledge model and user warning interface.

The extractor algorithm is used to extract the required feature on the current website. The knowledge model is used to compare the extracted feature to determine if the website is phishing, suspicious or legitimate. The user warning interface has three modules, and the first is the voice generation with text directive with a red colour status if the requested site is a phishing web page. The second is the text direction with voice generation and amber colour status if the requested site is suspicious, while the third is the voice generation with text directive with green colour status if the requested site is legitimate page. The plugin is implemented in MATLAB version 9.5 AppDesigner toolbox. The online toolbar was tested and evaluated with 1000 phishing websites, 100 suspicious and 1500 legitimate websites. The average results achieved is 93.28% accuracy, which indicates a higher performance with the use of hybrid features of image, text and frame. These issues are covered in more detail in chapters 4 and 5.

3.7 Classification in Machine Learning

Decision Trees (C4.5 Algorithm)

A decision tree is an approach that can be used for prediction and classification (Machado and Gadge, 2017). An example of the use of a decision tree is phishing classification, where the phishing dataset is provided to determine the actual category each data belong either legitimate or phishing website. Usually, a good analyst seldom needs to ask all the questions to get to the correct answer. In phishing website classification, the decision tree acts as the analyst and represents a series of questions, where the answer to the first question determines the next question to be asked and so on. In the making of a decision in the algorithm, the features of interest are entered in the root node, and then branches for each possible value of these features are built. This procedure is used recursively until all the features in the root node end up in the same class or the tree is unable to split into any further branches (Machado and Gadge, 2017). The categorisation of the various features is a crucial task because it affects the distribution of the classes in each division. The decision tree algorithm procedure can be employed in various ways to build any scheme. When the tree has been built, each route from the source node to each of the end nodes characterises a rule. The behaviour of the rule is driven by the route from the source node to the end node, and the result is the standard class that is allocated by the end node. Various trimming approaches can be used to streamline the rules and to remove needless ones. Trimming the tree involves either substituting some sub-trees with end nodes or moving some nodes upwards to swap them with nodes that are higher up in the tree and more advanced (Abdelhamid, Thabtah and Abdel-jaber, 2017). In both processes are examples of post-trimming techniques use to make an algorithm to classify data more accurately (Verma and Rai, 2015).

The Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm

The repeated incremental pruning to produce error reduction (RIPPER) algorithm is a rule-training algorithm that was developed by William Cohen (Abdelhamid, Thabtah and Abdel-jaber, 2017). The RIPPER algorithm works as follows: First, the training dataset is separated into two sets: a pruning set and a set with the most considerable reduction of error. The RIPPER then develops its classifier using these two sets by constantly injecting rules at the initial stage from a new ruleset. The rule-generating algorithm starts with a clear rule and then heuristically adds one condition at a time until the rule has no error rates when applied to the growing set. First, a new stopping criterion for creating rules has been introduced in RIPPER. The algorithm stops injecting rules by following the minimum description length (MDL) principle when after a rule is injected into the algorithm, the total categorisation length of the rules set, and the training data is predicted. If the description length is higher than the lowest MDL obtained so far, RIPPER stops inserting rules. The MDL adopts the top perfect set of rules and data that is the one with reduces the size of the scheme and the quantity of essential information for recognising exemptions that are related to the scheme (Abdelhamid, Thabtah and Abdel-jaber, 2017).

The Prism Algorithm

The prism algorithm can be described as a covering algorithm for building classification rules. A covering algorithm begins by taking one class from among the existing ones in the training dataset and then seeks a way of covering all the instance in that class, while at the same time eliminating those that are not in that class. This approach is used to attempt to create rules with the best accuracy by injecting one condition at a time into the existing rule behaviour. The prism algorithm chooses at each iteration the condition that makes the most of the probability more desired for classification. The procedure for building a rule stops as soon as a stopping criterion is met. Prism

continues to build once a rule is derived, and the rules for the current category until all instances related to the class are achieved. After this occurs, another class is designated, and so on. The prism algorithm usually produces seamless rules with a 0% error rate and calculates the accuracy of its rules using an accuracy formula.

Without taking any influence on any existing rules and work independently, the prism algorithm has an advantage over the decision tree in that, in the former, a rule can be injected into the created rule set at a later point in time, whereas adding a route to a tree structure may necessitate a redesign of the whole tree (Verma and Rai, 2015). Nevertheless, unlike decision trees that classify an instance by using rules formed and reading them directly from the tree, the prism algorithm might suffer from some problems because of the separateness of the rules, where, for example, an instance may turn out to be related to more than one rule and different classes.

The Projective Adaptive Resonance Theory (PART) Algorithm

In contrast to the RIPPER and decision tree algorithms which both have a two-phase approach to the creation of rules, the projective adaptive resonance theory (PART) algorithm creates rules one at a time and thus avoids the comprehensive pruning phase (Cohen, Nissim and Elovici, 2018). The decision tree algorithm employs a divide-and-overcome approach to set rules, while the RIPPER algorithm uses a separate-and-overcome tactic to derive rules. On the other hand, the PART algorithm combines both of these methods to find and produce rules. It employs the separate-and-overcome approach to create a set of rules and uses the divide-and-overcome approach to develop partial decision trees. However, while the PART algorithm develops and trims a partial decision tree in the same way as a decision tree does, rather than build a whole decision tree, it instead develops some partial decision trees. The PART algorithm also differs somewhat from the RIPPER algorithm in terms of how it creates rules. In the PART algorithm, each rule links to the end node with the most extensive coverage in the partial decision tree, whereas in the RIPPER algorithm the

rules are built in an acquisitive manner; the RIPPER algorithm starts from a clear rule and adds condition until the rule has no error rate and then this process is repeated for other rules.

Neda Abdelhamid, Aladdin Ayesh, Fadi Thabtah (2017) conducted experimental tests using the decision tree, RIPPER and PART algorithms on different datasets and showed that notwithstanding the ease of using PART, it can produce sets of rules that are as accurate as those produced by the decision tree and that are more accurate though more complex than those of the RIPPER algorithm.

The Classification Based on Association (CBA) Algorithm

Classification and association rule detection are two of the essential tasks in data mining. Association mining is used to discover descriptive information from datasets, while classification focuses on developing a classification scheme for labelling new data. However, both design detection and classification association rule mining are essential to real-world data mining applications. If these two related tasks could be combined somehow, this would result in huge savings time and greater convenience for the user. Hence, substantial efforts have been made to combine these two methods into one system. In recent years, a comprehensive study has been carried out to integrate both techniques (Abdelhamid, Thabtah and Abdel-jaber, 2017).

Association rule mining and classification are analogous tasks, with the exception that the critical objective of classification is the prediction of class labels, whereas the main aim of association rule mining is to describe the relationship between feature values in a dataset. In the last few years, association rule mining has been used effectively to develop an accurate classifier, which has resulted in a new technique known as associative classification (AC) (Tripathi, Nigam and Edla, 2017). Associative classification is a branch of data mining that combines classification and association rule mining.

The AC algorithm utilises association rule detection approaches in the classification of datasets. Numerous studies (Jeeva and Rajsingh, 2016; Tripathi, Nigam and Edla, 2017; Wang, Kannan and Ulmer, 2013) have indicated that AC techniques can extract more accurate classifiers than legacy classification approaches such as the decision tree (Machado and Gadge, 2017) and (Abdelhamid, Thabtah and Abdel-jaber, 2017) and probabilistic (Ramanathan and Wechsler, 2012) approaches.

The CBA algorithm was the first to use association rules for feature classification (Abdelhamid, Ayesh and Thabtah, 2014). This algorithm produces a unique subset of association rules called class association rules (CARs). The variance among the association rules and CARs denotes the significance of the rules. The significance of the CARs is only restricted to the class label value. Therefore, the process of CAR that is called rule-item is $X \rightarrow C$, where C is a set of all class labels. There are two parts to the CBA algorithm: a classifier constructor named CBA-CB and a rule producer named CBA-RG. In the CBA-RG part, all frequent rule-items are created by using a procedure that is like that used in association rule mining. In the CBA-CB, all recurrent rule-items from the CBA-RG are graded in decreasing the order of importance.

A training dataset contains the activities used in the procedure of selecting the rules for the classifier; the algorithm repeats the process through each rule beginning with the first-order rule to find all the transactions comprising all features in the behaviour of the existing rule. Although at least if the rule covers one activity is classified effectively with this rule, the rule is nominated into the classifier, and all these activities enclosed in the rule is removed from the dataset, else the rule is trimmed. This procedure ends when either all of the rules are used, or no transactions are left in the dataset. Hence, a default class is designated by the favourite class in outstanding transactions.

Support Vector Machine (SVM) Algorithm

In recent times, Support Vector Machine (SVM) is one of the popular classifiers used for dataset classification and label them in different categories. The purpose of using the SVM is to locate the optimal separating hyperplane between two or more classes by locating the closest points between the maximum margin. Assume we have a linear discriminating function and two phishing features with a target value -1 and +1. The discriminating hyperplane will be determined by:

$$s' + s_0 \geq 0 \ if \ t_i = +1; \quad (3.12)$$

$$s'x_i + s_0 < 0 \ if \ t_i = -1 \quad (3.13)$$

Then the distance of any point x to hyperplane is $| s'x_i + s_0 | / || s ||$ and the distance to the starting point is $| s_0 | / || s ||$.

The SVMs are very powerful and commonly used in classification, but they suffer some drawbacks. The algorithm uses high computational power to train the dataset. Also, the algorithm is sensitive to the noisy dataset and subdue to overfitting.

The above algorithms were considered in our study using them to conduct the various experiment and discover that the SVM and KNN have better performance than other compare to ANFIS and the deep learning algorithm the CNN-LSTM. Both SVM and KNN was used as a reference to compare the results of this study in chapter 4.

3.8 Chapter Summary

This chapter presented a description of the research methodology, including the choice of feature extraction and analyses methods based on appropriate literature sources. The extracted data from all the sources used in this study were used as input including the image, frame and text features that were identified as being the most important for ongoing phishing detection. The chapter also described the units of analysis

comprising the sets of inputs that were chosen to give a reasonable representation of phishing methods and approaches, and later five combinations of parameters were used in the parameter tuning framework. Out of the total number of sources, including documentary sources that were classified, the proposed system was able to achieve a 98.3% success rate in the identification of legitimate and illegitimate websites. The system should restore online user confidence in their activities online. The work does not raise any ethical concerns about personal information as no features about individuals that might enable their recognition were collected.

Some of the work in this chapter has been published by the author as part of his doctoral research; the full citation is: Adebowale, M. A., Lwin, K. T., Sánchez, E. and Hossain, M. A. (2019) 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text', *Expert Systems with Applications,* 115, pp. 300-313.

Chapter 4 will present the implementation of the system and the experimental setup and results.

CHAPTER 4: IMPLEMENTATION OF INTELLIGENT PHISHING DETECTION SYSTEM (IPDS)

Chapter 4 Implementation and Evaluation of an IPDS

In the previous chapter, research methodology was present that describes features-based online and offline approaches for phishing detection. The chapter also includes the sources selection method and features extraction. Features were analysed, and the advantages and the disadvantages are considered, various machine learning approaches were also studied and the ones with relatively good performance were chosen for adoption. In the proposed model, images, frames and text features are used to improve the detection of a phishing website. The key challenge is to develop an intelligent plugin using the ANFIS algorithm combined with a knowledge model and feature inputs. To accomplish this objective various phishing detection algorithms, learning design and feature extraction techniques have been explored for phishing website detection as described in chapter 3. A smart system is improved by using a supervised machine learning algorithm. The cross-validation will involve k-fold or holdout using the ANFIS and another classifier to train and test the phishing detection system on the website features. The entire detection system can then be deployed as a web browser plug-in. Although the existing text-based features approach applying machine learning techniques developed elsewhere (Barraclough *et al.*, 2013; Barraclough, Sexton and Aslam, 2015) the attempts to detect phishing website still have error rates that need to be addressed. Also, work using visual similarity and text still suffer from errors (Haruta, Asahina and Sasase, 2017; Zhang *et al.*, 2011). The solution proposed in this current work combines all three elements of the website, the image, and the frame and text to improve accuracy and reduce error rates. In addition, this chapter presents the system architecture and experimental procedures from the methodology described in the previous chapter that includes training and testing with validation results.

Fig. 4-1 is the modified form of an online phishing detection system architecture described elsewhere (Barraclough, Sexton and Aslam, 2015) that illustrates feature

extraction based on hybrid feature inputs (sources) which include image features of the website, frame that includes the source code, and tags and cascaded style sheet (CSS). This also includes the text content of the website. These three hybrid input features were used to extract 35 comprehensive features. These features are utilise hybrid and adaptive neuro-fuzzy inference system (ANFIS) machine learning techniques and fuzzy rules during training and in online plugin system development.

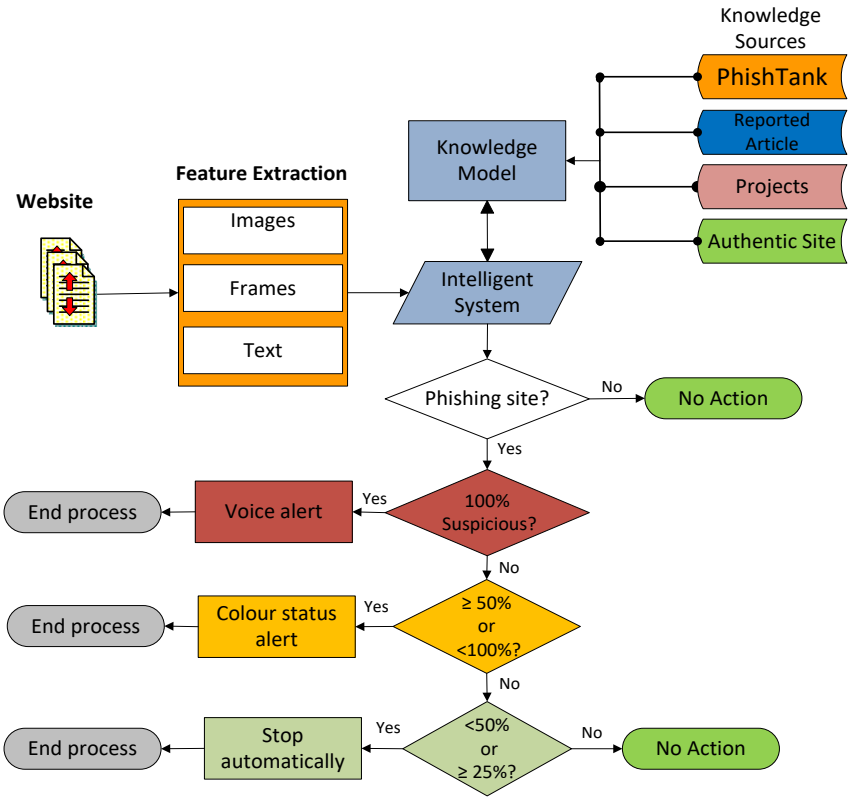


Fig. 4-1: Intelligent Phishing Detection System Structure (Barraclough, Sexton and Aslam, 2015)

In this case, these techniques were used to classify if a website is legitimate, suspicious or phishing in real-time. If a phishing website is detected, a sound alarm is generated to alert the user. If the site is phishing, a red colour status is activated together with a text-based risk explanation to inform the user of the threat. Also, if the site is suspicious, that if more than 50% (but less than 100%) of the content is dubious, the amber colour status is activated with text directive alerting the user. Hence, in a

situation where the ruleset is violated, a warning is generated to alert the user. If the threat is less severe i.e. between 25% and 50% then the user can continue. However, if it is highly likely that sensitive information will be stolen, then the process is automatically stopped, and the fake website does not acquire the user's confidential information. The system is presented in Fig. 4-1 above.

4.1 Intelligent System

Intelligent phishing detection solutions have been proposed in recent years that include text-based approaches (Aburrous *et al.*, 2010; Barraclough *et al.*, 2013; Zhuang, Jiang and Xiong, 2012) but need the additional robustness provided by utilising additional frame and image features. One such approach is to use a web browser for phishing detection that incorporates a toolbar (Aggarwal, Rajadesingan and Kumaraguru, 2012; Dunlop, Groat and Shelly, 2010; Ghosh, 2013; Kalola, Patel and Pandit; Microsoft, 2015) incorporating anti-phishing icon functionality such as the blocking the source. Another approach involves checking visual similarity such as using the EMD algorithm to predict whether website legitimacy (Zhang *et al.*, 2011; Kumar and Kumar, 2015b; Haruta, Asahina and Sasase, 2017).

On the other hand, a server-side solution approach involves a two-factor verification which ensures that users know the secret OTP before they are allowed to continue using the website. In the current model parameter optimisation methods are assigned to the hybrid features (backward propagation and least-squares) using the ANFIS algorithm to determine the level intelligent detection, supplemented later by a deep learning algorithm to improve the robustness of the scheme.

4.2 Knowledge Model

Knowledge is one of the primary requirements of decision-making. Using a knowledge model for phishing detection is an excellent idea as it leads to practical and efficient system development. In this research, the knowledge model will be incorporated into

the classification and prediction process. The model will consist of an expert system and official websites (Aggarwal, Rajadesingan and Kumaraguru, 2012). These two elements of the system design will enable quick verification of a website's legitimacy. It is anticipated that the validity provided by the knowledge model will enable the system to perform in an efficient manner that will reduce the time needed by the proposed intelligent phishing detection scheme (IPDS) to perform its prediction task. This is system may be described as IPDS (Fig. 4.1) which the knowledge model is stored in an Excel sheet format that is updated every 5 minutes.

4.3 Conceptual Framework Using ANFIS

The overall conceptual block diagram of the proposed intelligent phishing detection system is presented in Fig. 4-2.

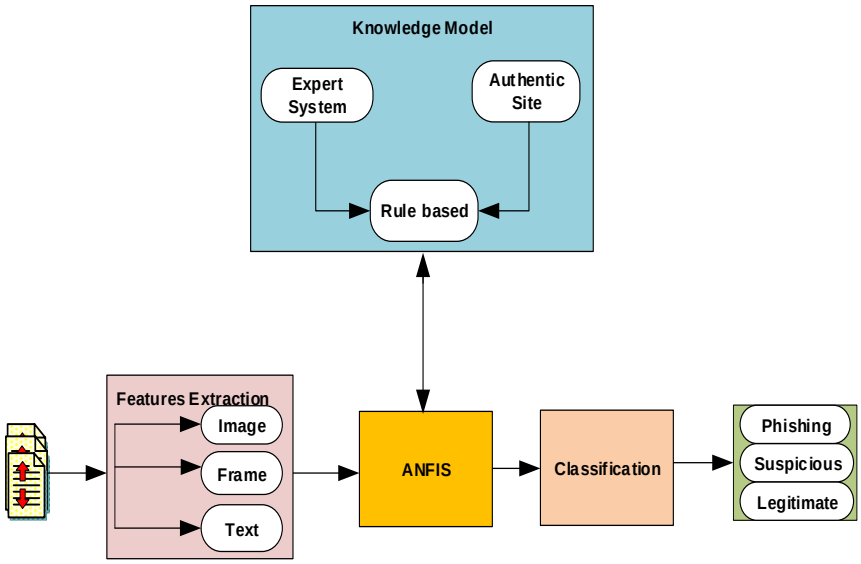


Fig. 4-2: Conceptual Block Diagram of IPDS based on Image, Text and Frame Features

This block diagram illustrates the process of acquiring the website features and feeding them into the fuzzy inference system for training, testing and validation purposes. Then the ANFIS using fuzzy IF…THEN rules are applied to distinguish legitimate, suspicious and phishing websites accurately in real-time. The conceptual block diagram of the classification process of the proposed model, which is a modification of the paths

defined in Barraclough *et al*. (2013) is presented in Fig. 4-3. For this study the classification will utilise k-fold cross-validation the entire feature dataset for training and the testing.

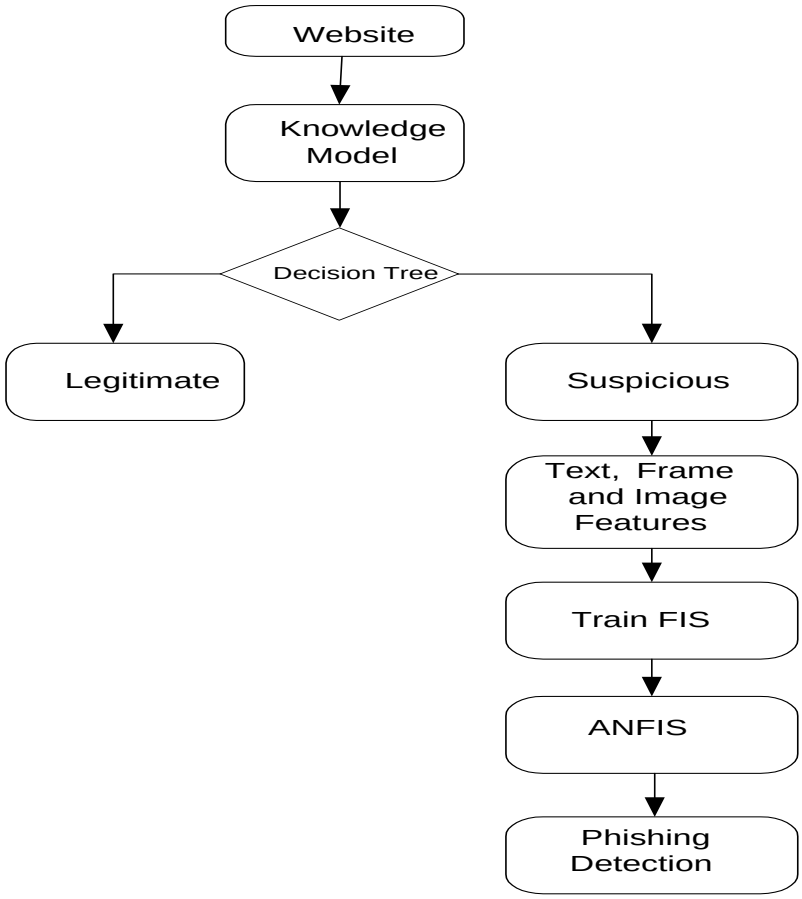


Fig. 4-3: Conceptual Block Diagram of Intelligent Phishing Website Detection Classification

4.3.1 ANFIS Dataset

The dataset was collected between April 2016 and May 2018, and the datasets had 35 attributes for a total of approximately 2,456 website hits and a total number of 13,056 datasets sample (Adebowale, 2019). The features were assigned weights, where the value of the legitimate sites which had little risk was assigned a weight of 0.1; those that had a medium risk and considered suspicious were assigned a weight of 0.3, and those with a high risk that were judged to be phishing sites were given a weight of 0.6.

***Advantage*:**

The extraction of features is a significant part of this research as they are used to create our models and produce ***If-then*** fuzzy rules to train and test the scheme. Furthermore, these features are also used in the online toolbar to keep track of current websites against the set of features in order to classify them as phishing, suspicious or legitimate sites accurately in real-time.

Disadvantages:

Phishing methods are enhanced frequently, so this may put pressure on the available features list as needs to be continuously updated to remain relevant and accurate. To address this issue, an automated system is developed that allows the phishing knowledge model to be updated every 5 minutes when new phishing sites are added to the PhishTank and some other sources, after which an Excel file is created to store the update as a query with the function to perform a search to return the returning variables. However, it should be noted that the Excel query function cannot simplify a new phishing phrase until an expert manually identifies it because such phrases evolve frequently.

4.3.2 CNN+LSTM Dataset

The proposed IPDS uses two DL layers to classify phishing websites by applying LSTM on the text and frame content and the CNN on the images. Thus, the model can easily explore the richness of the words embedded in the website's URL as well as the images on the site. The performance of the proposed model was tested by applying it to a phishing dataset that consisted of one million URLs taken from PhishTank and a legitimate site Common Crawl as well as over 10,000 images from legitimate and phishing websites that were personally collected from various e-commerce and e-banking sites over some time. This dataset was used to train and test the network

using holdout cross-validation; 70% and 30% of the dataset was used for these purposes, respectively.

4.4 ANFIS Experiment Setup

During the training phase, the algorithm learns and extract features from the data file, reads them and uses them to create fuzzy rules (Abunadi, Akanbi and Zainal, 2013). Various methods have been put forward in the literature for the training of fuzzy models. These include 2-fold (Barraclough et al., 2013) cross-validation, 5-fold cross-validation, 10-fold cross-validation and 20-fold cross-validation. In this study, the 5-fold cross-validation method was employed to train and test the accuracy and also the robustness of the algorithm. The five-fold cross-validation procedure involves dividing the dataset into five parts, four for training and one for testing. The five-fold cross-validation method was chosen due to its effectiveness for common datasets (Barraclough, Sexton and Aslam, 2015). The five-component input parameter also includes three membership functions. Also, the generalised bell-shape membership function was chosen for various ranges on the x-axis and y-axis. In order to improve the model's effectiveness and overcome the problems of a long operational time and a high false-positive rate, a hybrid method for parameter optimisation was applied using 10 epochs and a zero-tolerance error set (Aburrous *et al.*, 2010).

The neuro-fuzzy designer toolbox and classification learner toolbox in MATLAB Version 9.5 were used for the experiment. The parameters in the neuro-fuzzy designer (Fig. 4-4) were set as follows: the number of epochs was set to 10, the error rate was set to 0, and the optimisation method was set to hybrid, and also, various cross-validation has used the results were presented in Table 4-5.

Of the 35 features derived from the experimental datasets, 22 were text-based, 8 were frame-based, and 5 were image based. The full datasets were divided into 7 separate sets with 5 features in each set. Each of these sets were loaded into the input

layer in random order for training. During the training phase, the errors were corrected by back-propagation. The inputs are processed through the inference systems and the neural network, with the rules provided to decide throughputs to the output layer and this was repeated in such a way that the dataset is used once. The error rates are recorded in Table 4-6.

After the training phase had been completed, the testing phase was conducted. The testing phase followed the same procedure as the training phase. The actual decision process at the authentication stage is the last output by which the system can compare the features to decide if the site is phishing, suspicious or legitimate. Here too the procedure was also repeated for each input dataset such that the datasets were only used once. The average error rate was calculated by summing up the error rates and then dividing by the number of data, and this rate was used to measure the performance of the proposed algorithm (Table 4-6).

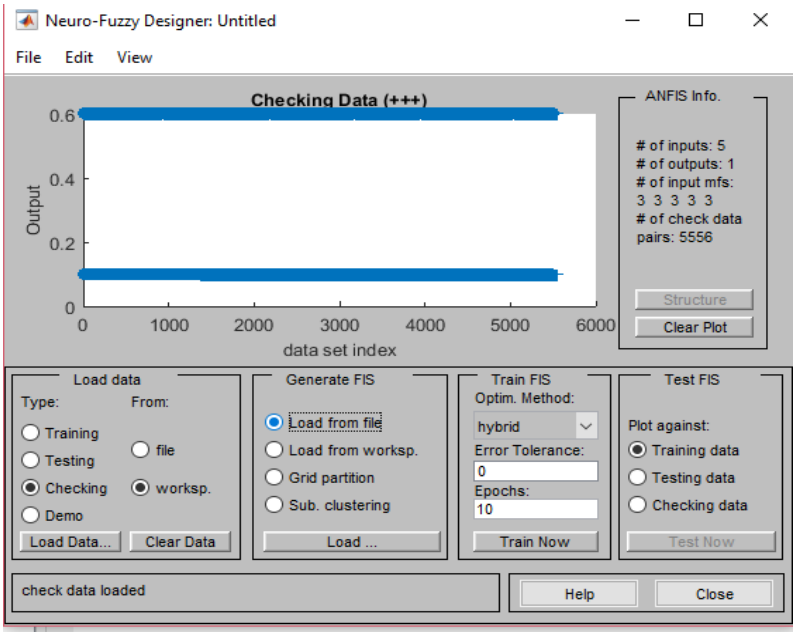


Fig. 4-4: Parameter Settings Used for Neuro-Fuzzy Designer (Source: own)

In this study, an SVM and KNN was also used to validate the result of the hybrid feature selection process. The SVM has already been used efficiently in the classification and detection of phishing emails, text and deceptions (Huang, Qian and

Wang, 2012). The SVM can target a useful training dataset that allows accurate classification of small training sets (Adewumi and Akinyelu, 2016). Hence an SVM was used in this study to validate further the hybrid features utilised in the experiment. Moreover, KNN was used as an instance-based learning algorithm because it is beneficial for a variety of problem domains in which the original scores are not known (Babu, Nirmala and Kumar, 2010). Furthermore, KNN has been applied to text classification since the early days of data mining research and has been shown to produce a better result than a variety of other machine learning algorithms (Jain and Gupta, 2016b).

The quadratic SVM and KNN in the MATLAB version 9.5 classification learning toolbox was explicitly used to compare the model. The toolbox was loaded with the 35 features that were extracted in this study. The features used for training and testing and the result that was obtained were tabulated using a confusion matrix analysis (Fig. 4-5). The results of the training and testing are presented in Table 4-6. In addition, to validate the proposed approach, the results were compared with those of other machine learning algorithms Tables 4-1 to 4-4.

4.4.1 Performance Measure

A performance measurement in feature classification is essential in assessing the quality of the learning model. The performance measurement can be defined as the procedure of quantifying the accuracy and efficiency of an action using a metric that is associated with the action. Various measures have been described in the literature that can be used to make an improved decision overall or a particular area of application (Gupta and Shukla, 2015). In line with the above, an evaluation of the various metrics was undertaken, which identified the following measures to be used:

True Positive Rate:

The True Positive (TP) rate denotes the percentage of phishing websites that are correctly classified as a phishing site or the percentage of URLs that were correctly classified as a phishing URL. The higher the percentage, the better the performance of the classifier (Xiang *et al.*, 2011). This measure is calculated using the following formula:

$$TruePositive = \left(\frac{TP}{(TP+FP)}\right) X\ 100\% \qquad (4.1)$$

False Negative Rate:

The False Negative (FN) rate represents the percentage of phishing websites/URLs that are classified wrongly as a legitimate website/URLs. The lower the percentage the improve is in the classifier performance (Hassan, 2015). This measure is calculated as follows:

$$FalseNegative = \left(\frac{FN}{(TP+FN)}\right) X\ 100\% \qquad (4.2)$$

True Negative Rate:

The True Negative (TN) rate denotes the percentage of legitimate websites/URLs that are classified incorrectly as a phishing site or URL (Xiang *et al.*, 2011). This rate is calculated by:

$$TrueNegative = \left(\frac{TN}{(TN+FN)}\right) X\ 100\% \qquad (4.3)$$

False Positive Rate:

The False Positive (FP) rate indicates the percentage of legitimate websites/URLs that are classified correctly as a phishing site/URL. The lower the percentage, the better the performance of the classifier (Hassan, 2015). This rate is calculated by using the formula below:

$$FalseNegative = \left(\frac{FP}{(TP+FP)}\right) X\ 100\% \quad (4.4)$$

Accuracy:

The accuracy measurement is the simplest and most common measure used to evaluate a classifier (Pradeepthi and Kannan, 2014). The accuracy of a model is denoted by the percentage of correct predictions it makes or by the percentage of misclassification errors it makes. The following formula calculates the accuracy of the proposed detection and classification model:

$$Accuracy = \frac{(TP+TN)}{(TP+FP+TN+FN)} \quad (4.5)$$

Precision:

Precision is the proportion of correct classifications made by the classifier. If there are a large number of classifications into a given category where many should not be there then this lowers the precision (Pradeepthi and Kannan, 2014). The precision is calculated by the following:

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (4.6)$$

The precision measure is what machine learning, data mining and information retrieval research studies focus on primarily, but this measure is entirely ignored in receiver operation characteristics (ROC)[6] analysis (Powers, 2011).

Recall:

The recall contrasts with the precision in recording the number of correct classifications as a proportion of all positive classifications included the miss-classified (false positive) outputs. The recall is calculated by the following:

[6] In statistics, a receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied [Online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Receiver_operating_characteristic [Accessed on 15 December 2016].

$$Recall = \frac{TP}{(TP+FN)} \qquad (4.7)$$

F-Measure:

The F-measure is the harmonic mean of recall and precision. The F-measure is calculated by:

$$F - Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \qquad (4.8)$$

The F-measure effectively references the TP to the arithmetic mean of the predicted positives and the actual positives in proportion to a specific agreement in the actual class and the set-Dice coefficient (Powers, 2011).

Although a confusion matrix integrates all the performance measures of the classification algorithm, additional meaningful results can be extracted from the matrix to reveal the performance measures Fig 4-5.

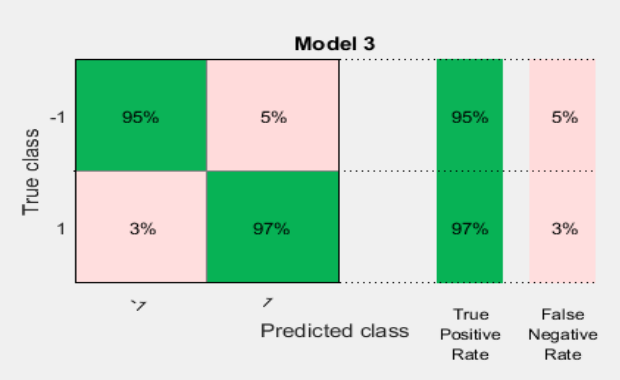


Fig. 4-5: Confusion Matrix for Phishing Dataset

4.5 Discussion of the Results and Analysis of ANFIS Experiment

The purpose of cross-validation was to examine the overall performance of the three inputs on multiple training and testing standard machine learning techniques. This section presents descriptive statistical results for the intelligent phishing detection model experiments. The results of the experiments are shown in Tables 4-1 to 4-4 below, derived from the confusion matrix data in conjunction with the performance

measures. All measures are presented as percentages. The overall average training accuracy was 98.55% for text features, 98.06% for frame features, 97.2% for image features and 98.3% for hybrid features.

Table 4-1: Classification Result Using Text Features

Algorithm	Accuracy %	Recall %	Precision %	F-measure %
ANFIS	98.55	98.51	98.58	98.54
KNN	95.50	95.45	95.54	95.49
SVM Quadratic	94.30	94.29	94.31	94.29

Table 4-2: Classification Result Using Frame Features

Algorithm	Accuracy %	Recall %	Precision %	F-measure %
ANFIS	98.06	98.02	98.08	98.02
KNN	59.59	59.20	59.60	59.39
SVM Quadratic	59.99	59.90	60.10	59.99

Tables 4-1 and 4-2 show that the KNN and SVM have a low accuracy result compared to the ANFIS algorithm, for a detection time of 52.6 seconds.

Table 4-3: Classification Using Image Features

Algorithm	Accuracy %	Recall %	Precision %	F-measure %
ANFIS	97.20	97.18	97.22	97.18
KNN	59.20	59.19	59.21	59.20
SVM Quadratic	63.30	63.29	63.32	63.30

Table 4-4: Classification Using Hybrid Features

Algorithm	Accuracy %	Recall %	Precision %	F-measure %
ANFIS	98.30	98.26	98.31	98.28
KNN	96.10	96.05	96.14	96.09
SVM Quadratic	95.20	95.18	95.23	95.20

Table 4-5: ANFIS Cross-Validation

ANFIS Cross-Validation Data Source Result				
	Image	Text	Frame	Hybrid
2-fold	96.71	93.49	95.44	94.23
5-fold	97.20	98.55	98.06	98.2
10-fold	95.03	91.35	95.1	92.53

Table 4-6 presents the result of applying the ANFIS 5-fold cross-validation method with five features as input. The first column lists the seven feature sets. The Test error column presents the total testing errors derived from the standard testing error results for each feature set. The next three columns collectively headed Training error presents the total training errors derived in a similar way, summarized as a Training error% in the next column. The average Training error can be seen to be 1.7%. The final column shows that the model had an overall training accuracy of 98.3% using 5-fold cross-validation with an average time of 26.72 seconds. Note that the ANFIS stops learning when the tolerant limit of testing error is reached (Karaboga and Kaya, 2016).

Table 4-6: ANFIS 5-fold Cross-Validation Method with Five Feature Input

Result summary for five inputs	**Test error**		**Training error**		**Training error%**	**Training average error%**	**Training accuracy**
Feature set 1	0.022388	0.019986	0.015759	0.014006	1.49%		
Feature set 2	0.014926	0.013324	0.015092	0.012795	1.39%		
Feature set 3	0.034624	0.03662	0.016522	0.013896	1.52%		
Feature set 4	0.014936	0.013324	0.014206	0.016406	1.53%		
Feature set 5	0.02063	0.03331	0.013387	0.01294	1.32%	1.7%	98.3%
Feature set 6	0.014628	0.02662	0.016828	0.021965	1..94%		
Feature set 7	0.032621	0.03662	0.021453	0.034447	2.80%		
Average error	0.023897		0.017122				

The results of the proposed scheme were compared with the approach proposed by Abdelhamid *et al.* (2014), which used multi-label classifier-based associative classification (MCAC) to produce an 94.5% accuracy. It was also compared with the method suggested by Barraclough et al. (2013) for phishing detection using neuro-fuzzy, which obtained 98.55% accuracy (Chart 4-1) for text-only feature detection. However, the present experiment included fine-tuning of the features arranged together in the same attack pattern for training and testing and assigning different weights with a reduction in some functions by removing the redundant elements used in their model.

Chart 4-1: Experimental result for ANFIS, SVM and KNN classification

4.6 ANFIS Limitations

Even though the ANFIS is the most popular algorithm for feature-based fuzzy modelling, it still has some limitations. The most common one relates to the input type membership function in that it can only be either constant or linear. Therefore, future

work should identify other possible parameter combinations to expand the framework to make it more effective.

The two most important conclusions to draw from this ANFIS approach are:

- The ANFIS is a first order Sugeno fuzzy model. The ANFIS is a NN with five layers: input, fuzzification, fuzzy rule, normalisation and defuzzification (Çakıt and Karwowski, 2017).
- The ANFIS utilises a hybrid learning algorithm that aggregates the result of the least-squares estimator by using the gradient descent method. A training set of inputs is presented in the forward pass, after which, neuron outputs are calculated for each layer. The subsequent parameter rules are identified by using a least-squares estimator, while the error signals are propagated backwards and the antecedent parameter rules are updated according to the chain rule (GüNeri, Ertay and YüCel, 2011).

4.7 Conceptual Framework Using Deep Learning

The overall conceptual framework for the intelligent phishing detection system (IPDS) uses deep learning as presented in Fig. 4-6. The concept involves using two deep learning algorithms, namely LSTM and CNN on different types of features that have been extracted from websites in order to better predict phishing activities. The feature extraction step and machine learning are applied in the initial stage in the classification process. A block diagram of the proposed anti-phishing detection system is shown in Fig. 4-7.

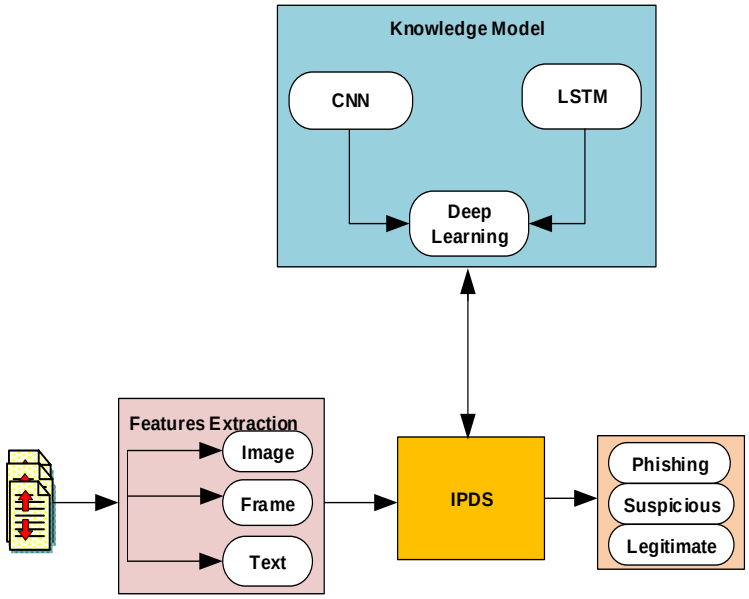


Fig. 4-6: Conceptual Block Diagram for Intelligent Phishing Detection System (IPDS)

The block diagram of the IPDS in Fig. 4-6 above illustrates the process of acquiring the website features and feeding them into the deep learning system for classification purposes. Then, the trained LSTM-CNN network is applied to distinguish accurately between legitimate, suspicious and phishing websites in real-time. Websites are assessed separately to ascertain whether they are legitimate or fake (phishing). The feature set used 70% for the training set and 30% for the testing set.

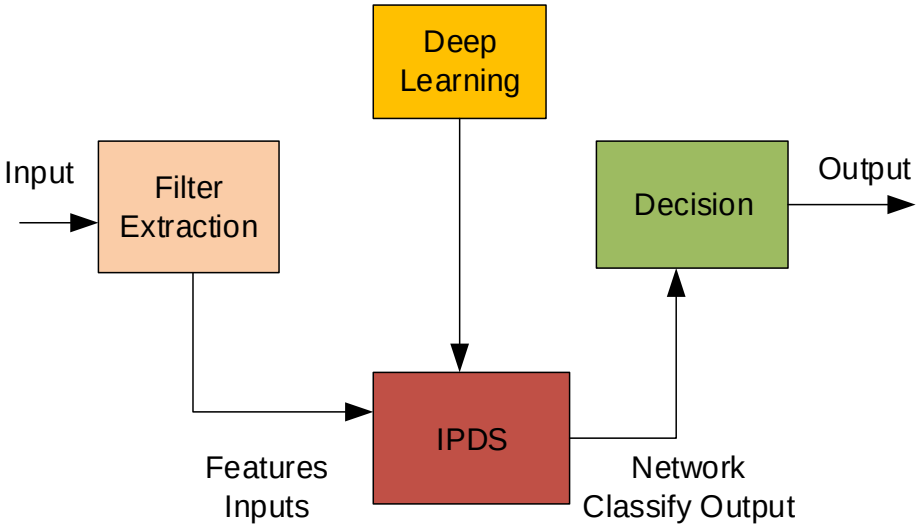


Fig. 4-7: Block Diagram of Intelligent Phishing Detection System (IPDS) Structure

4.7.1 LSTM+CNN Experiment Setup and Results

The raw data from both the images and the URLs contained a lot of background information and varied in length and size. Therefore, pre-processing was necessary to make the data available for training the model. For the CNN architecture, images from the sites were cropped based on the springing-box and merely removed the wrong image. For the LSTM architecture, several URLs were collected and saved in Microsoft Excel format as comma-separated-values with only the URL in one column and their category label in the other column as shown in chapter three (see Table 3-7).

The model was developed in MATLAB version 9.5 using the deep learning toolbox. For the CNN architecture, there were three categories of data. The AlexNet CNN was used, which is eight layers deep and can classify an image into over 1,000 object categories. The network has a wide range of images as well as many learned rich features. The AlexNet network has an input image size of 227-by-227. In order to take advantage of the architecture the pre-trained network was retrained with the images obtained from various websites for the network so that it would be able to classify new images. The AlexNet network was edited using the MATLAB deep learning toolbox. Pre-trained learning and the fully connected layer output size were changed to a three-fold classification of legitimate, suspicious and phishing categories. Both the bias learning rate factor and the learning rate factor were set 10. The first classification layer was deleted, and the new layer was connected. The newly connected classification layer was analysed, and the report showed zero errors. The new network was then exported into the deep network design. After that, the extracted image dataset was loaded into the image data storage and processed to extract the speeded-up robust features (SURF) from all the images using the grid method to create a bag of features where the data was split into 70% for training and 30% for validation using holdout cross-validation. The images were resized to match the sizes of those in the pre-trained network input. In order to train the network, the exported edited AlexNet

network layer from the toolbox were used to train image and set the options. Then clustering was used to create a 1000-word visual vocabulary (Fig. 4-8). The model took 130 sec to complete one epoch of the training procedure.

For the LSTM architecture, the dataset was partitioned, and holdout cross-validation was set to 70% for training and 30% for validation. The URLs were tokenised to separate each URL into a series of separate words, all of which were set in lowercase. The tokenised data was then encoded to make it available for training, where the maximum length was set to 75, the hidden size was set to 180, and the embedding dimension was 100 with the fully connected network. The training options were set (Fig. 4-9) to *adam*; epoch = 10, gradient threshold = 1, learning rate = 0.01 and verbose = false. By doing this, the network architecture layer was modified to achieve better training accuracy.

Fig. 4-8: Training Dataset Word Vocabulary (Source: Own)

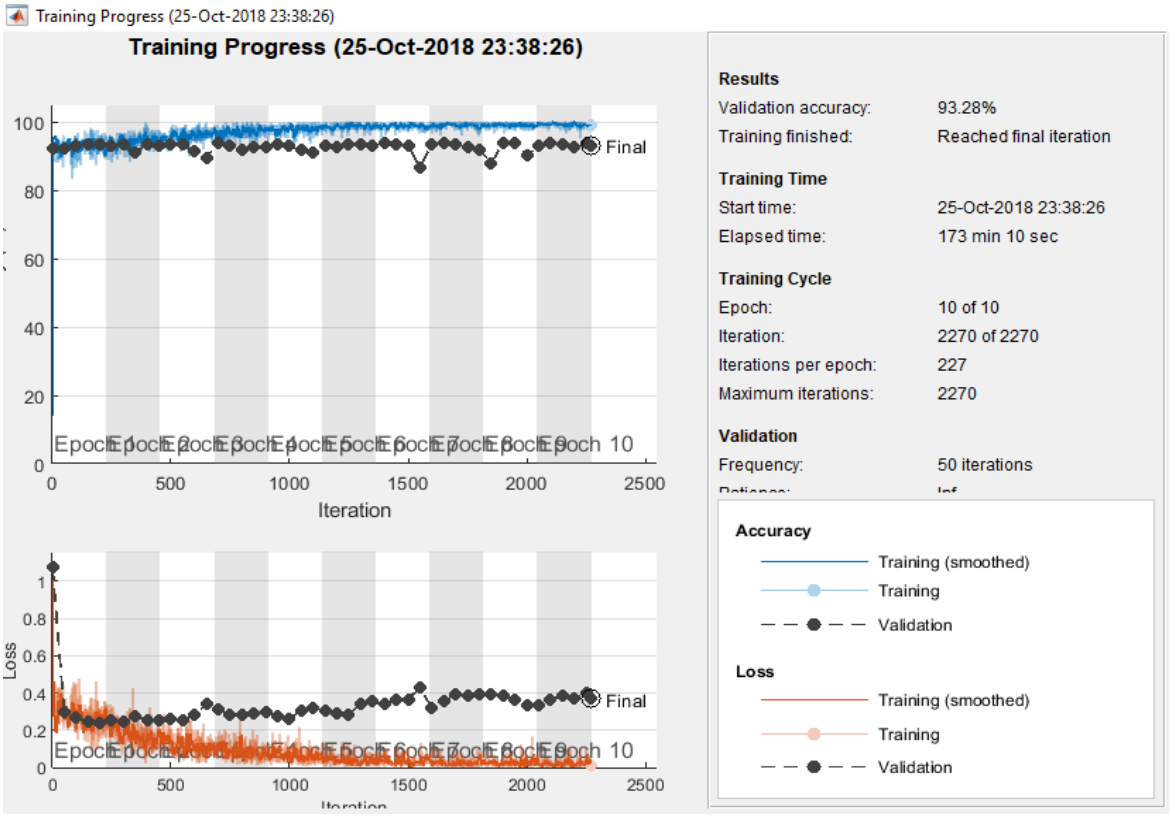


Fig. 4-9: CNN-LSTM Training and Validation Process (Source: Own)

The evaluation of the proposed method was performed based on traditional feature engineering, plus the classification algorithm methodology presented in section 4.8. As described earlier the total average training accuracy of the scheme was 93.28% (Table 4-7). The training achieved relative performance in CNN with 92.55% and that for testing was achieved by LSTM with 92.79% (Chart 4-2). The results showed that the average accuracy of the model was high, at 93.28%. The results show that some level of improvement in phishing detection was achieved through the use of hybrid features by combining the images, text and frames of a site with the use of a hybrid DL algorithm.

The results also provide information about the usefulness of unsupervised pre-training and the effectiveness of image feature extraction in detecting phishing sites.

Table 4-7: Classification Result for CNN, LSTM and IPDS (CNN+LSTM)

Algorithm	Accuracy %	Recall %	Precision %	F-measure %
CNN	92.55	92.51	92.58	92.54
LSTM	92.79	92.78	92.81	92.80
IPDS (CNN+LSTM)	93.28	93.27	93.30	93.29

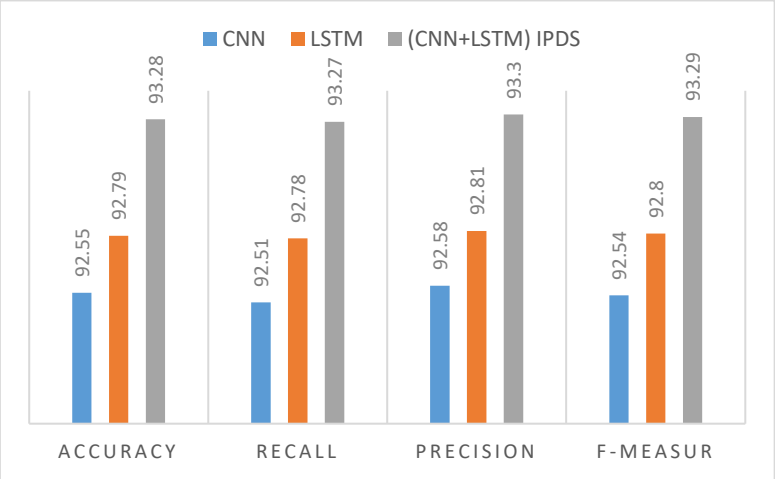


Chart 4-2: Experiment Result for Deep Learning by CNN, LSTM and IPDS (CNN+LSTM)

4.8 Chapter Summary

In this chapter hybrid features from diverse sources are discussed, the ANFIS and the CNN-LSTM algorithm was used for experimental setup. The study reflects the effectiveness of the hybrid features approach using ANFIS and CNN, and the LSTM deep learning algorithm is an essential driver to the high model performance. This chapter has contributed to the anti-phishing detection research by present the use of a hybrid feature which include image, frame and text. These three sets of input have just been introduced as single hybrid features for the first time. The three elements are used because they represent the whole structure of a website. Although the scheme performed well, parameter tuning influenced the algorithm in a positive way and it must

be pre-specified to solve a given problem. Ultimately online user confidence will increase in performing transactions online.

Note that some content of this chapter has been reported by the author in *Expert Systems with Applications*, *Journal of Enterprise Information Management* and IEEExplore conference paper, namely>

Adebowale, M. A., Lwin, K. T., Sánchez, E. and Hossain, M. A. (2019) 'Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text', *Expert Systems with Applications,* 115, pp. 300-313.

M. A. Adebowale, K. T. Lwin and M. A. Hossain, "Deep Learning with Convolutional Neural Network and Long Short-Term Memory for Phishing Detection," *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, Island of Ulkulhas, Maldives, 2019, pp. 1-8.

M. A. Adebowale, Lwin, K. T. and Hossain, M. A. (2020) 'Intelligent phishing detection system using deep learning algorithm', *Journal of Enterprise Information Management,*

CHAPTER 5: IMPLEMENTATION AND EVALUATION OF PHISHING DETECTION TOOLBAR

Chapter 5 Implementation and Evaluation of Phishing Detection Toolbar

5.1 Introduction

In the previous chapter, the hybrid feature-based offline scheme was developed and tested using the adaptive neuro-fuzzy inference system (ANFIS) optimisation approach with a deep learning convolution neural network (CNN) utilising long short-term memory (LSTM). The section of the extraction and the dataset was briefly described. The experimental procedures and results were also presented, analysed and discussed.

The evaluation of the proposed method was performed based on traditional feature engineering, plus the classification algorithm methodology presented in section 4.8. Features were created based on the URLs, image features and website elements. The CNN and LSTM classifier were trained using one million URLs and over 10,000 images to build the model. A Toolbar concept was developed using a deep learning (DL) algorithm against legitimate, suspicious and phishing websites. The results showed that a voice-generating user warning interface with a green colour status and a text showing a warning was generated within 25 seconds before the page loaded to give the user a warning.

5.2 System Architecture Design and Theoretical Definitions

The MATLAB version 9.5 AppDesigner toolbox was used to create a graphical user interface to evaluate the model. The checking process involved the user entering the URL link into the textbox. When the check button is pressed, the colour of the traffic light changes to correspond to the classification of the URL and the text also displays the classification value. Fig. 5-1 shows the result for a phishing site; Fig. 5-2 shows the result for a suspicious site and Fig. 5-3 shows the result for a legitimate site.

5.3 Time-Based and Accuracy-Based Tests

In this experiment, standard assessment metrics were applied to assess the performance of the developed detection toolbar for phishing websites using real-time-based and accuracy-based evaluation methods (Xiang *et al.*, 2011). The time-based method explores performance under conditions that are similar to those found in a real-world situation, while the accuracy-based method assesses overall performance on the available dataset. Both of these methods were implemented in order to evaluate the new method rigorously.

Due to the advances in technology and the adoption of new techniques, phishers have been able to improve their forged websites so that they now have high similarity with legitimate sites in terms of content. In tests, the current state-of-the-art solutions have been able to obtain 70% to 98% accuracy (see Table 2-1) in identifying legitimate website. However, these solutions must perform well in the real world, so there needs to be a significant improvement of 0.5% or higher (Shirsat, 2018). Moreover, their level of accuracy in identifying suspicious websites should be higher still, and their accuracy in detecting phishing websites should be even higher (Government Communication Headquarter. (GCHQ, 2018).

For the present research, three series of experiments were performed for each evaluation method, testing them against legitimate, suspicious and phishing websites. In the time-based evaluation process, the time-to-classify point was recorded against all the legitimate datasets, suspicious datasets and phishing datasets. Then the process was also repeated several times to determine the average time to each classification at an interval. In the accuracy-based assessment, all the legitimate datasets, suspicious datasets and phishing datasets were utilised to test the toolbar.

The accuracy of the model was tested using the holdout cross-validation strategy. In that experiment, the overall classification accuracy result (Chart 5-1) for the

proposed IPDS (CNN+LSTM) was 93.28% (Table 5-1). The classification achieved a relative performance in CNN with 92.55% and that for testing was achieved by LSTM with 92.79% (Table 5-1).

Table 5-1: Relative Performance of CNN, LSTM and IPDS (CNN+LSTM)

Algorithm	Accuracy %	Recall %	Precision %	F-measure %
CNN	92.55	92.51	92.58	92.54
LSTM	92.79	92.78	92.81	92.80
IPDS	93.28	93.27	93.30	93.29

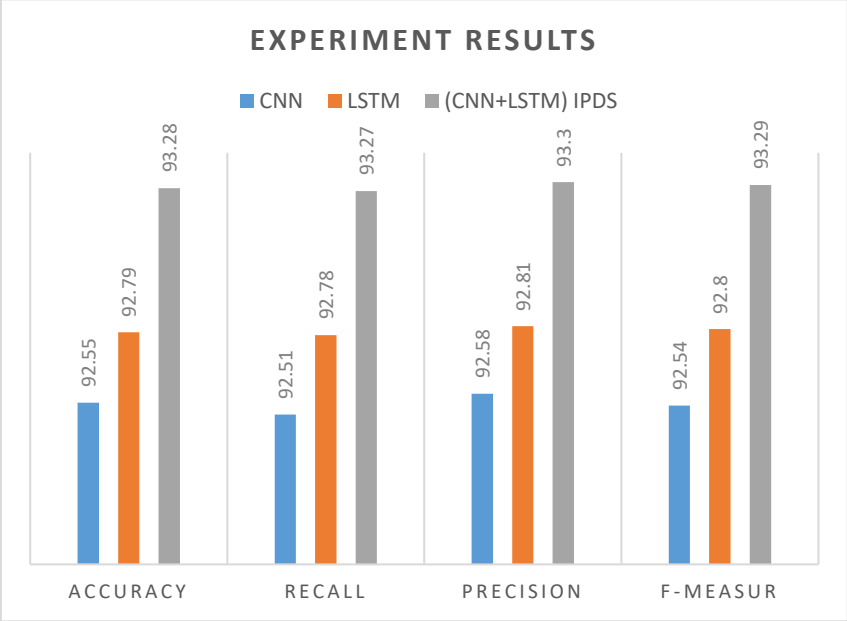


Chart 5-1: Relative Performance of Algorithms in Chart Form

5.3.1 Testing of the Toolbar Application on Phishing Websites

To evaluate the toolbar concept, it was tested on 2,600 websites including legitimate, suspicious and phishing websites. First, it was tested on 1,000 phishing websites. The LSTM-CNN algorithm runs in the background as a knowledge module. When a URL is typed into the address bar (Fig 5-1), the algorithm inspects whether the requested website is a phishing link by comparing the current URL against the stored features in

the deep learning classification algorithm. If a match is detected, and it is a phishing site, in order to alert the user a red colour status with a voice-operated user warning interface is activated and a text is generated showing that the status of the URL is “phishing”.

The above procedure was repeated up to 1000 times with different URLs, so all the phishing URLs were tested. The performance of the toolbar in each case was observed and recorded, and besides, screenshots were taken to validate the results. An example of a screenshot of a phishing website result is shown in Fig 5-1. The rest of the screenshots are presented in Appendix D. This part of the experimental effort was carried out over 8 hours per day for five consecutive days. As regards the time-based assessment of the toolbar’s ability to detect a phishing website, the voice-generating user warning interface with a red colour status and a text showing an alert were generated within 25 seconds to warn the user before the page loaded.

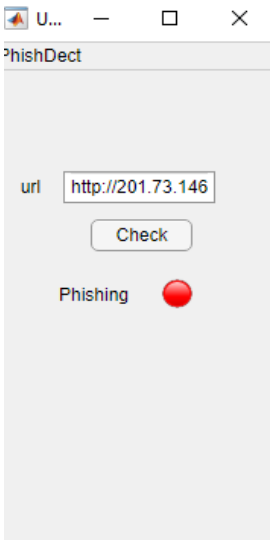


Fig. 5-1: Testing of the Application on Phishing Websites

5.3.2 Testing of the Toolbar Application on Suspicious Websites

The toolbar also evaluated on 100 suspicious URLs. As previously mentioned, the LSTM-CNN algorithm runs in the background as a knowledge module. The same

procedure is followed as in the testing of the toolbar on phishing websites that described in the previous section, but in this test, the algorithm checks whether the URL requested is a suspicious website by relating the newly typed URL against the stored features in the IPDS. If a match is detected, and it looks like the URL is a suspicious website, the user warning interface included in the model shows an amber colour status and, besides, a text description is generated stating that the URL is “suspicious” (Fig. 5-2) in order to alert the user to exercise caution. This process was repeated 500 times on all 100 URLs and the performance was observed and recorded (Table 5-2). An example of a screenshot of suspicious website results shown in Fig. 5-2. The rest of the screenshots are presented in Appendix E This task required 8 hours per day over two days to perform because the finding shows that there is a little and a reasonable number of suspicious online websites which make this challenging task as they are short-lived. As regards the time-based assessment of the toolbar’s performance in identifying a suspicious website, the voice-generating user warning interface with an amber colour status and a text showing a warning were generated within 25 seconds to alert the user before the page loaded.

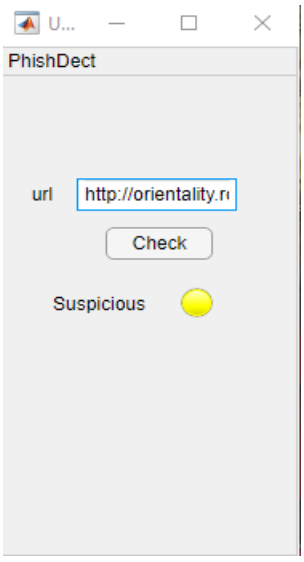


Fig. 5-2: Testing of the Application on Suspicious Websites

5.3.3 Testing of the Application on Legitimate Websites

The toolbar was also tested on 1,500 legitimate URLs. As stated above, the LSTM-CNN algorithm runs in the background as a knowledge module. The same procedure as that used to test the toolbar's performance on phishing and suspicious websites was used, but in this instance, the algorithm checks whether the URL that has been requested is a legitimate website by relating the newly typed URL in text box against the stored features in the IPDS. If no match is found, then it is a legitimate website, and the user warning interface displays a green colour status (Fig. 5-3). At this point, it is safe for the user to continue in their task with peace of mind that the site to which they are submitting their confidential information is legitimate.

In the experiment, this procedure was repeated 600 times with validation dataset consisting of URLs so that most the URLs were tested to validate the performance of the toolbar and in each case, the result was observed and recorded (Table 5.2). Figure 5-3 shows an example of a screenshot of one of the results produced by the toolbar for a legitimate site. The rest of the screenshots are provided in Appendix F. As regards the time-based assessment of the toolbar's ability to detect a legitimate website, the voice-generating user warning interface with a green colour status and a text showing the result was generated within 25 seconds before the page loaded. The next section presents an evaluation of the toolbar's performance in terms of accuracy and speed.

Fig. 5-3: Testing of the Application on Legitimate Websites

5.4 Evaluation of the Accuracy-based and Time-based Results

The most essential and standard evaluation metrics used for this online phishing website detection plugin experiment is the real-time based feature and the accuracy-based evaluation method (Bayani, 2013). This section focuses on the accuracy and speed at which the proposed phishing detection toolbar works, because not only is the accuracy of the warning important, the warning interface needs to be able to alert users quickly in real-time.

On the one hand, the inclusion of a voice-generating user warning interface, as well as text directives, made a significant improvement to the effectiveness of the proposed application. While the accuracy-based method is measuring the performance of the on the available dataset, the time-based method is to explore the performance of the plugin on a real-world scenario. On the other hand, the experiments showed that the overall performance of the intelligent phishing detection system was active at both the time-based and accuracy-based level. Suitable testing was done to assess the overall performance of the intelligent phishing detection.

The existing solutions obtain between 70% to 90% accuracy on the testing with the phishing dataset. For this reason, three series of experiments are conducted for each

evolution of the method, validating the scheme against phishing, suspicious and legitimate datasets.

5.4.1 Validating of Performance

In order to validate the performance of the plugin, a total of 1,000 phishing websites were used to evaluate the performance of the plugin in terms of time taken and accuracy. Overall, the toolbar was able to achieve an average accuracy of 93.28%, as shown in Table 5-2. Then in Table 5-2 column 4 roll 2, shows the performance of the phishing detection with 93.8% true positives and in column 5 roll 2, 6.2% true negative this has taken into consideration using 1000 phishing URLs with an accuracy of 93.5% in column 3 roll 2. Also, the toolbar achieved 94.5% accuracy shown on column 3 roll 3, with 94.8% true positives column 4 roll 3and 5.2% true negative in column 5 roll 3 when tested on 100 suspicious datasets. Meanwhile, when the plugin is tested on 1,500 legitimate websites, the phishing detection toolbar achieved 91.8% accuracy column 3 roll 4, was recorded with true positives of 92% column 4 roll 4 and 8% real negative in column 5 roll 4. However, accuracy varies from a minimum of 91% to a maximum of 94%, which caused significant variation in the accuracy results across the testing datasets.

Table 5-2: Test Results for Phishing Website Detection by Toolbar Application

Status	No. of Websites	Accuracy %	TP%	TN %	Average Result %
Phishing websites	1000	93.5%	93.8%	6.2%	93.28%
Suspicious websites	100	94.5%	94.8%	5.2%	
Legitimate websites	1500	91.8%	92%	8%	

5.4.2 Time-Based Performance

In order to confirm the scheme on time-based performance, the toolbar was validated against the legitimate, suspicious and phishing websites. The average time for a website to load its content is typically 60 seconds as reported by one researcher on his system (Aburrous *et al.*, 2010). This load time allowed the tool bar plug in to be designed to schedule the detection in various stages to present a decision in real-time (see Table 5-3). The table reflects the first check of the plugin to appear in 10 seconds, checking for most used features of the phishing website at the initial stage. Hence, if any of these features were found on the web page, the user is alert accordingly. Then the plugin goes to the second stage to verify the site against the list features in the second stage in Table 5-3, which the result should be decided within 25 seconds. The third stage is the graphic checking state which takes the most time to load, and so was scheduled to be the last check. When a phishing website was requested, the plugin alerted the user within 25 seconds before the interface loaded its result using a voice-operated user warning interface with a red colour status and a text showing the status of a phishing website. Compared to the time needed to detect a phishing site, the plugin needed a more extended period to determine whether a URL was a link to a legitimate website. The plugin took an average running time of 30 seconds before the interface loaded its result, showing that the URL was a legitimate site. This is because phishing URLs have some unique features that can be easily identified by the scheme.

Based on the above results of testing and validating the design concept on legitimate, suspicious and phishing websites, this study has demonstrated the accuracy of the scheme using the proposed algorithm. To the user's knowledge this study is the first to consider the LSTM-CNN algorithm for use in phishing detection and the first to use a comprehensive set of features that includes image, test and frame content from all possible sources from a broad spectrum of websites.

Table 5-3: Real-time detection stages

First level check					Time
Using the IP Address	Long URL to Hide the Suspicious Part	Using URL Shortening Services	URL's having "@" Symbol	Redirecting using	10s
Adding Prefix or Suffix Separated by (-) to the Domain	SubDomain and Multi SubDomains	HTTPS: HYPERText Transfer Protocol with Secure Sockets	Using Non-Standard Port	The Existence of "HTTPS" Token in the Domain Part of the URL	
Second level check					
Domain Registration Length	Request URL	URL of Anchor	Submitting Information to Email	Abnormal URL	25s
Age of Domain	DNS Record	Website Traffic	PageRank	Google Index	
Number of Links Pointing to Page	Statistical-Reports Based Feature	IFrame Redirection	Server Form Handler (SFH)	Using Pop-up Window	
Tags	Disabling Right Click	Website Forwarding	Layout Similarity	Style Similarity	
Third level check					
Favicon	Image Size	Alternative Text	Mouse over	Login Form	15s

The approach presented in this study is not directly comparable with existing works because none of those works uses all the possible features of the image, frame and text content in terms of size and range. The majority of the previous studies used precise elements of websites such as blacklists and text features to develop anti-phishing toolbars (Bottazzi *et al.*, 2015). For example, Sharma, Meenakshi and Bhatia (2017) surveyed ten toolbars and found that the existing toolbars mostly use text features and blacklists. Among the existing toolbar that uses URL features and blacklists are shown in section 2.4, Table 2-1 and 2-2 with an average of 94%.

5.5 Chapter Summary

The results of this research demonstrated that using the hybrid features of a website with some directives can alert users accurately, effectively and in real-time to the presence of phishing websites. Moreover, using a wide range of different features can

improve the false negative error rate. This is in line with existing research in phishing detection that has found that URLs alone do not represent all of the characteristics of phishing techniques (Li *et al.*, 2019). Also, the fact that legitimate sites had a disparity in detection accuracy was because some spam was malicious and contributed to the false positives.

This study demonstrates that the fundamental requirement for accurate performance in phishing detection is a combination of comprehensive features, ANFIS, a deep learning algorithm and a warning system that can be activated in real-time. The new form of the user interface toolbar approach is a novel contribution to existing knowledge and could be extended/generalised to detect malicious emails and/or developed into a commercial product.

Finding many live unknown websites for testing was an unexpectedly difficult issue as the life span of phishing websites is typically only 48 hours. However, 100 suspicious websites were used as representative examples. It is essential to clarify that the features that were used in this research were up-to-date and active at the time of writing. However, phishing techniques evolve rapidly, so regular updating with new phishing characteristics is recommended to ensure that the system remains accurate.

Some of the results presented in this chapter have been reported by the current researcher journals and conference paper.

CHAPTER 6: CONCLUSION & FUTURE WORK

Chapter 6 Conclusion & Future Work

6.1 Concluding Remarks

Phishing is a significant problem that leads to identity theft, and it requires an efficient and proactive solution. Although phishing attacks are often simple in design, phishers are very active, and attacks are becoming more complex, so they have caused millions of pounds' worth of damage in recent years. Due to the growing severity of the problem this technically challenging and academic area was chosen as the focus of this research which aimed to extend the current approaches to phishing detection by using the adaptive neuro-fuzzy inference system (ANFIS) algorithm.

However, there are two main issues associated with using ANFIS which this current research addresses. First, it is more complicated because it must have a single output obtained by using weighted average defuzzification. Second, all the output, whether constant or linear, must have the same membership function (Barraclough *et al*., 2013). Another issue prior to the current work is that no recent studies have used text, frame and image features together to automatically detect phishing websites in real-time.

6.2 Research Achievement

This study presents an intelligent phishing detection and protection scheme (IPDS) that was developed by employing a new approach using the integrated features of images, frames and text of phishing websites mentioned in the objective.

The aim of the study was to use an efficient ANFIS algorithm to develop the offline IPDS, tested and verified for phishing website detection and protection. Also to meet the set object various experiment were conducted, and the results validate that the proposed approach was able to achieve an accuracy of 98.3% (See Table 4-6) which, at the time of writing, is the best available integrated solution for web-phishing

detection and protection, this work being presented by the author in two of the Tier One journal publications, namely 'Expert System with Application' (ESWA) and published in 2019 and *Journal of* Enterprise Information Management in 2020.

The offline real-time approach using ANFIS was able to classify a phishing website in an average of 30 seconds less than the average time of 60 seconds (Barraclough *et al.*, 2013) for a web browser to load on the user system, and thus is a practical and viable add-on tool for browser functionality, which was part of the aim of the research.

This study also explored the efficacy of the deep learning approach, which is part of the set objective to explore relevant algorithm for the detection of phishing, this revealing the advantages and disadvantages of both the convolutional neural network (CNN) and long short-term memory (LSTM) methods. On the one hand, the LSTM+CNN algorithm was also used to develop an offline approach for phishing detection but had a smaller detection accuracy of 93.28% compared to that of the ANFIS algorithm.

The LSTM+CNN algorithm performed faster than the ANFIS algorithm in the classification of phishing and legitimate websites with an average of 25 seconds. On the other hand although it is faster the prediction performance is on average slightly lower than that of ANFIS. Given the faster performance the LSTM+CNN model was later used to develop a browser-plugin this fulfilling one of the researches aims and objectives.

The plugin was tested with over 2,000 websites reporting an average detection accuracy of 93.28% within 25 seconds in real-time. The results of using the LSTM+CNN algorithm have been presented at the conference of Software Knowledge Information Management and Application (SKIMA 2019), 26th to 28th August 2019 in the Island of Agulhas, Maldives. The work was also later published in IEEEXplore along with an extended version as reported earlier, in 2020 in the Journal of Enterprise Information Management.

The dataset collated for use in this current work has been made available online, at: “Adebowale, M. A. (2019) 'Phishing Detection Dataset' (Version 2) [Numeric Data]’ and accessible at: http://dx.doi.org/10.17632/gt7xdbs3kt.1 e.g. accessed: 25 November 2019).

6.3 Contribution to Knowledge

The primary contribution of this study is the integration of hybrid features from text, images and frames that were extracted from various websites and then used to develop a robust ANFIS solution. First, a hybrid feature selection approach for use in the detection of website phishing attacks was developed. The method is based on a combination of content-based and visual-based approaches. The hybrid feature selection approach uses legitimate and non-legitimate websites and an associated artificial intelligence algorithm to develop an integrated method to address these elements and is referred to in this work as the Intelligent Phishing Detection and protection Scheme (IPDS). The hybrid approach used a dataset containing one million legitimate and phishing websites from the PhishTank and Common Crawl datasets to validate the scheme as well as 10,000 images that were collected from both phishing and legitimate websites. The 10,000 images and the 13,000 features dataset were used to build the knowledge model, which has been placed in the public domain as a resource for other researchers to develop their own contributions to knowledge in the field of phishing detection solutions,

The features from a previous study (Barraclough *et al.*, 2013) were optimised from over 300 features in their solution. One significant achievement of the present solution is that it reduces the number of features in their text model to 22 by removing the redundant ones and including 8 frame features and 5 image features for a better optimised solution of 35 hybrid features in the current study. The reduction in the number of features makes this much faster in terms of time-to-prediction. The protection aspect of the solution is implemented via a user warning interface with

various colours representing the category of detection. A green colour indicates a legitimate site, whilst an amber colour represents suspicious ones, and a red colour indicates a phishing site. There is also an audible (voice) warning of relevance to a visually impaired person. The protection interface also advises the user on what to do next such as to terminate the process if it discovers that the site is phishing or suspicious.

The main conclusion of applying the IPDS approach that is proposed in this study is the achievement of an excellent classification accuracy of 93.28% for identifying phishing websites. Previous chapters discussed the success in enhancing both the offline model and the online toolbar for phishing detection. In particular a MATLAB solution uses a voice generating user warning interface based on 35 features set to detect a phishing web page in real-time. The real-time approach is one of the strengths of this study. This was demonstrated in the previous chapter 5, with the IPDS able to respond with great agility and could verify a legitimate web page in 30 seconds. To the best of the current author's knowledge, the research presented is the first work that considers how best to integrate image, text and frame features into a combined solution for a phishing detection scheme.

6.4 Future Work

The current work has established some areas that could provide further directions of research. These are described as follows;

- There needs to be further study to evaluate what other variables could be used to improve classification accuracy and reduce the false-positive rate of the classifiers.
- Since the detection system is automated, there needs to be further work as to how the knowledge base might be kept up to date to reflect new trends in phishing attacks. In other ways, how the protection system can be dynamic in

response to changing phishing attack strategies (such as use of non-standard ports, as described in chapter 5). The current non-dynamic limitation of the present solution is the main barrier to the intelligent solution being able to achieve maximum accuracy and optimum performance.

- The approach would benefit from a better validation of the features, such as 'Abnormal DNS Record' and 'Abnormal Request URL' possibly using appropriate database queries on listings of known reputable registered and lawful websites.
- The algorithm for determining the similarity of website features needs to be improved, as the current system does not base comparisons on standardised content (text, frame, images, style, javascript), that is, there is a need to standardise content so that like-for-like comparisons can be made.
- The toolbar plug in needs to be developed so that it is cross-browser compatible and able to be used by all standard browsers (Internet Explorer, Mozilla Firefox, Google Chrome). It could also be developed as a Desktop application, possibly running in the background for spontaneous phishing detection as well as self-directed tool for specific phishing detection.

Reference

Abbasi, A., Zahedi, F. M., Zeng, D., Chen, Y., Chen, H. and Nunamaker, J. F. (2015) 'Enhancing Predictive Analytics for Anti- Phishing by Exploiting Website Genre Information', *Journal of Management Information Systems,* 31(4), pp. 109-157.

Abbasi, A., Zhang, Z., Zimbra, D., Chen, H. and Nunamaker Jr, J. F. (2010) 'Detecting fake Websites: the contribution of statistical learning theory', *Mis Quarterly*, pp. 435-461.

Abdelhamid, N., Ayesh, A. and Thabtah, F. (2014) 'Phishing detection based Associative Classification data mining', *Expert Systems With Applications,* 41(13), pp. 5948–5959.

Abdelhamid, N., Thabtah, F. and Abdel-jaber, H. 'Phishing detection: A recent intelligent machine learning comparison based on models content and features', *International Conference on Intelligence and Security Informatics (ISI)*, Beijing, China, 22-24 July 2017: IEEE, 72-77.

Abraham, D. and Raj, N. S. 'Approximate string matching algorithm for phishing detection', *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, New Delhi, India, 24-27 Sept. 2014: IEEE, 2285-2290.

Abunadi, A., Akanbi, O. and Zainal, A. 'Feature extraction process: A phishing detection approach'. *13th International Conference on Intellient Systems Design and Applications in Communications and Network Security (CNS) 2013*: IEEE, 331-335.

Aburrous, M., Hossain, M. A., Dahal, K. and Thabtah, F. (2010) 'Intelligent phishing detection system for e-banking using fuzzy data mining', *Expert Systems with Applications,* 37(12), pp. 7913-7921.

Abutair, H. Y. and Belghith, A. (2017) 'Using Case-Based Reasoning for Phishing Detection', *Procedia Computer Science,* 109, pp. 281-288.

Adebowale, M. A. (2019) 'Phishing Detection Dataset' (Version 2) [Numeric Data]. Available at: http://dx.doi.org/10.17632/gt7xdbs3kt.1 (Accessed: 25 November 2019).

Adewumi, O. A. and Akinyelu, A. A. (2016) 'A hybrid firefly and support vector machine classifier for phishing email detection', *Kybernetes,* 45(6), pp. 977-994.

Afroz, S. and Greenstadt, R. 'PhishZoo: Detecting Phishing Websites By Looking at Them'. *Fifth IEEE International Conference on Proceedings of the Semantic Computing (ICSC)*, Palo Alto, CA, USA, 18 September 2011: IEEE, 368-375.

Aggarwal, A., Rajadesingan, A. and Kumaraguru, P. 'PhishAri: Automatic real-time phishing detection on twitter'. *eCrime Researchers Summit (eCrime)*, Las Croabas, Puerto Rico, 23-24 Oct. 2012: IEEE, 1-12.

Ahmed, A. A. and Abdullah, N. A. 'Real-time detection of phishing websites'. *7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Vancouver, BC, Canada, 13-15 Oct. 2016: IEEE, 1-6.

Al-Daeef, M. M., Basir, N. and Saudi, M. M. 'A Method to Measure the Efficiency of Phishing Emails Detection Features', *International Conference on Information Science & Applications (ICISA)*, Seoul, South Korea, 6-9 May 2014: IEEE, 1-5.

Aleroud, A. and Zhou, L. (2017) 'Phishing environments, techniques, and countermeasures: a survey', *Computers & Security,* 68, pp. 160-196.

Alkhozae, M. G. and Batarfi, O. A. (2011) 'Phishing websites detection based on phishing characteristics in the web page source code', *International Journal of Information and Communication Technology Research,* 1(6), pp. 283-291.

AlShboul, R., Thabtah, F., Abdelhamid, N. and Al-diabat, M. (2018) 'A visualization cybersecurity method based on features' dissimilarity', *Computers & Security,* 77(2018), pp. 289-303.

Amrutkar, C., Kim, Y. S. and Traynor, P. (2017) 'Detecting Mobile Malicious Web pages in Real Time', *IEEE Transactions on Mobile Computing,* 16(8), pp. 2184-2197.

Apruzzese, G., Colajanni, M., Ferretti, L., Guido, A. and Marchetti, M. 'On the effectiveness of machine and deep learning for cyber security'. *2018 10th International Conference on Cyber Conflict (CyCon)*, Tallinn, Estonia, 29 May-1 June 2018: IEEE, 371-390.

APWG (2019) *Unifying the Global Response to Cybercrime* [Online], Washington, D.C, USA: Anti-Phishing Working Group. Available at: https://docs.apwg.org/reports/apwg_trends_report_q2_2019.pdf.

Arachchilage, N. A. G. and Love, S. (2013) 'A game design framework for avoiding phishing attacks', *Computers in Human Behavior,* 29(3), pp. 706-714.

Arachchilage, N. A. G. and Love, S. (2014) 'Security awareness of computer users: A phishing threat avoidance perspective', *Computers in Human Behavior,* 38(2014), pp. 304-312.

Arachchilage, N. A. G., Love, S. and Beznosov, K. (2016) 'Phishing threat avoidance behaviour: An empirical investigation', *Computers in Human Behavior,* 60(2016), pp. 185-197.

Arel, I., Rose, D. C. and Karnowski, T. P. (2010) 'Deep Machine Learning - A New Frontier in Artificial Intelligence Research [Research Frontier]', *IEEE Computational Intelligence Magazine,* 5(4), pp. 13-18.

Armano, G., Marchal, S. and Asokan, N. 'Real-Time Client-Side Phishing Prevention Add-On'. *36th International Conference on Distributed Computing Systems (ICDCS)*, Nara, Japan, 27-30 June 2016: IEEE, 777-778.

Aydin, M. and Baykal, N. 'Feature extraction and classification phishing websites based on URL', *Conference on Communications and Network Security (CNS)*, Florence, Italy, 28-30 Sept. 2015: IEEE, 769-770.

Ba Lam, T., Luong Anh Tuan, N., Huu Khuong, N. and Minh Hoang, N. 'A novel fuzzy approach for phishing detection', *IEEE Fifth International Conference on Communications and Electronics (ICCE)*, Danang, Vietnam, 30 July-1 Aug. 2014: IEEE, 530-535.

Babagoli, M., Aghababa, M. P. and Solouk, V. (2018) 'Heuristic nonlinear regression strategy for detecting phishing websites', *Soft Computing*, pp. 1-13.

Babu, L. D. D., Nirmala, M. and Kumar, K. N. (2010) 'A Survey on Methodologies and Techniques for Detection and Prevention of Phishing Attacks', *International Journal of Advanced Research in Computer Science,* 01(03), pp. 152-159.

Bahnsen, A. C., Bohorquez, E. C., Villegas, S., Vargas, J. and González, F. A. 'Classifying phishing URLs using recurrent neural networks', *APWG Symposium on Electronic Crime Research (eCrime)*, Scottsdale, AZ, USA, 25-27 April 2017: IEEE, 1-8.

Bandhaniya, K. A. M. A. N. and Joshi, R. J. H. 'Survey Paper on a vulnerability in Relation to Phishing-Based Social Engineering Attacks'. *National Conference on Latest Trends in Networking and Cyber Security*, Ahmedabad Gujarat, India, March 2017: IJIRST, 149-153.

Barraclough, P. A., Hossain, M. A., Tahir, M. A., Sexton, G. and Aslam, N. (2013) 'Intelligent phishing detection and protection scheme for online transactions. (Report)', *Expert Systems With Applications,* 40(11), pp. 4697-4706.

Barraclough, P. A., Sexton, G. and Aslam, N. 'Online phishing detection toolbar for transactions'. *Science and Information Conference (SAI)*, London, UK, 28-30 July 2015: IEEE, 1321-1328.

Bayani, M. 'Web-based Decision Support Systems: A conceptual performance evaluation'. *IEEE 17th International Conference on Intelligent Engineering Systems (INES).* 19-21 June 2013, 21-26.

Baykara, M. and Gürel, Z. Z. 'Detection of phishing attacks', *6th International Symposium on Digital Forensic and Security (ISDFS)*, Antalya, Turkey, 22-25 March 2018: IEEE, 1-5.

Belabed, A., Aïmeur, E. and Chikh, A. 'A personalized whitelist approach for phishing webpage detection'. *Seventh International Conference on Availability, Reliability and Security (ARES), 2012*, Prague, Czech Republic, 20-24 Aug. 2012: IEEE, 249-254.

Bender, M., Childress, R. L., Kumhyr, D. B. and Spisak, M. J. (2018) *Detection of a Spear-Phishing Phone Call*. Google Patents. [Online].

Blum, A., Wardman, B., Solorio, T. and Warner, G. 'Lexical feature based phishing URL detection using online learning'. *Proceedings of the 3rd ACM workshop on Artificial intelligence and security*, Chicago, Illinois, USA, October 8, 2010: ACM, 54-60.

Bottazzi, G., Casalicchio, E., Cingolani, D., Marturana, F. and Piu, M. 'MP-Shield: A Framework for Phishing Detection in Mobile Devices'. *International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, Liverpool, UK, 26-28 Oct. 2015: IEEE, 1977-1983.

Breuel, T. M., Ul-Hasan, A., Al-Azawi, M. A. and Shafait, F. 'High-performance OCR for printed English and Fraktur using LSTM networks'. *12th International Conference on Document Analysis and Recognition*, Washington, DC, USA, 25-28 Aug. 2013: IEEE, 683-687.

Buber, E., Ö, D. and Sahingoz, O. K. 'Feature selections for the machine learning based detection of phishing websites', *International Artificial Intelligence and Data Processing Symposium (IDAP)*, Malatya, Turkey, 16-17 Sept. 2017: IEEE, 1-5.

Bullee, J.-W., Montoya, L., Junger, M. and Hartel, P. (2017) 'Spear phishing in organisations explained', *Information and Computer Security,* 25(5), pp. 593-613.

Chai, Z. and Jiang, H. 'A brief review on Decision Support Systems and it's applications'. *2011 International Symposium on IT in Medicine and Education (ITME).* 9-11 Dec. 2011, 401-405.

Chang, E. H., Chiew, K. L., Sze, S. N. and Tiong, W. K. 'Phishing Detection via Identification of Website Identity'. *2013 International Conference on IT Convergence and Security (ICITCS)*, Macao, China, 16-18 Dec. 2013: IEEE, 1-4.

Chauhan, S. and Shiwani, S. 'A honeypots based anti-phishing framework'. *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Kanyakumari, India, 10-11 July 2014: IEEE, 618-625.

Chen, T. C., Stepan, T., Dick, S. and Miller, J. (2014) 'An Anti- Phishing System Employing Diffused Information', *Acm Transactions On Information And System Security,* 16(4), pp. 1-31.

Cherdantseva, Y., Burnap, P., Blyth, A., Eden, P., Jones, K., Soulsby, H. and Stoddart, K. (2016) 'A review of cybersecurity risk assessment methods for SCADA systems', *computers & security,* 56, pp. 1-27.

Chiew, K. L., Chang, E. H., Sze, S. N. and Tiong, W. K. (2015) 'Utilisation of website logo for phishing detection', *Computers & Security,* 54, pp. 16-26.

Chin, T., Xiong, K. and Hu, C. (2018) 'Phishlimiter: A Phishing Detection and Mitigation Approach Using Software-Defined Networking', *IEEE Access,* 6, pp. 42516-42531.

Choo, X., Chiew, K., Ibrahim, D., Musa, N., Sze, S. and Tiong, W. (2016) 'FEATURE-BASED PHISHING DETECTION TECHNIQUE', *Journal of Theoretical and Applied Information Technology,* 91(1), pp. 101-106.

Churi, T., Sawardekar, P., Pardeshi, A. and Vartak, P. 'A secured methodology for anti-phishing', *International Conference on Innovations in Information, Embedded*

and Communication Systems (ICIIECS), Coimbatore, India, 17-18 March 2017: IEEE, 1-4.

CireşAn, D., Meier, U., Masci, J. and Schmidhuber, J. (2012) 'Multi-column deep neural network for traffic sign classification', *Neural networks,* 32(2012), pp. 333-338.

Cohen, A., Nissim, N. and Elovici, Y. (2018) 'Novel set of general descriptive features for enhanced detection of malicious emails using machine learning methods', *Expert Systems with Applications,* 110(2018), pp. 143-169.

Cohen, A., Nissim, N., Rokach, L. and Elovici, Y. (2016) 'SFEM: Structural Feature Extraction Methodology for the Detection of Malicious Office Documents Using Machine Learning Methods', 63, pp. 324-343.

Comar, P. M., Liu, L., Saha, S., Tan, P.-N. and Nucci, A. 'Combining supervised and unsupervised learning for zero-day malware detection'. *2013 Proceedings IEEE of INFOCOM*, Turin, Italy: IEEE, 2022-2030.

Cox, J. (2012) 'Information systems user security: A structured model of the knowing–doing gap', *Computers in Human Behavior,* 28(5), pp. 1849-1858.

Crain, J., Opyrchal, L. and Prakash, A. 'Fighting phishing with trusted email'. *ARES'10 International Conference on Availability, Reliability, and Security, 2010*, Krakow, Poland, 15-18 Feb. 2010: IEEE, 462-467.

Crosman, P., Quittner, J. and Wolfe, D. 2012. Bank of America Shows Anti- Phishing Leadership, But Much Work Remains. New York, N.Y.

Dadkhah, M., Shamshirband, S. and Abdul Wahab, A. W. (2016) 'A hybrid approach for phishing web site detection', *The Electronic Library,* 34(6), pp. 927-944.

Daeef, A. Y., Ahmad, R. B., Yacob, Y. and Phing, N. Y. 'Wide scope and fast websites phishing detection using URLs lexical features'. *3rd International Conference on Electronic Design (ICED)*, Phuket, Thailand, 11-12 Aug. 2016: IEEE, 410-415.

Dariane, A. and Azimi, S. (2016) 'Forecasting streamflow by a combination of a genetic input selection algorithm and wavelet transform using ANFIS models', *Journal of Hydrological Sciences,* 61(3), pp. 585-600.

Deshmukh, M. and Popat, S. K. (2017) 'Different Techniques for Detection of Phishing Attack', *International Journal of Engineering Science,* 7(4), pp. 10201-10204.

Deshmukh, M., Popat, S. K. and Student, U. (2017) 'Different Techniques for Detection of Phishing Attack', *International Journal of Engineering Science,* 7(4), pp. 10201-10204.

Dong, Z., Kapadia, A., Blythe, J. and Camp, L. J. 'Beyond the lock icon: real-time detection of phishing websites using public key certificates', *APWG Symposium on Electronic Crime Research (eCrime)*, Barcelona, Spain, 26-29 May 2015: IEEE, 1-12.

Dunlop, M., Groat, S. and Shelly, D. 'Goldphish: Using images for content-based phishing analysis'. *Fifth International Conference on Internet Monitoring and Protection (ICIMP)*, Barcelona, Spain, 9-15 May 2010: IEEE, 123-128.

eBay (2010) *Toolbar*. Available at: http://www.ebay.co.uk/sch/i.html?_nkw=ebay+toolbar (Accessed: 27 November 2016).

Fatt, J. C. S., Leng, C. K. and Nah, S. S. 'Phishdentity: Leverage Website Favicon to Offset Polymorphic Phishing Website'. *Ninth International Conference on Availability, Reliability and Security (ARES)*, Fribourg, Switzerland, 8-12 Sept. 2014: IEEE, 114-119.

Feroz, M. N. and Mengel, S. 'Phishing URL Detection Using URL Ranking', *International Congress on Big Data*, New York, NY, USA, 27 June-2 July 2015: IEEE, 635-638.

Financial Fraud Action, U. (2017) *Jan. to Dec. 2016 fraud update Payment cards, remote banking and cheque*. United Kingdom: Financial Fraud Action, UK. Available at: https://www.financialfraudaction.org.uk/fraudfacts17/.

Financial Fraud Action, U. (2018) *Jan. to Dec. 2018 fraud update Payment cards, remote banking and cheque*. London, United Kingdom: Financial Fraud Action, UK. Available at: https://www.financialfraudaction.org.uk/fraudfacts17/.

Gascon, H., Ullrich, S., Stritter, B. and Rieck, K. 'Reading Between the Lines: Content-Agnostic Detection of Spear-Phishing Emails'. *International Symposium on Research in Attacks, Intrusions, and Defenses*, Switzerland, 07 September 2018: Springer, 69-91.

Gaunt, R. E. (2016) 'The rate of convergence of some asymptotically chi-square distributed statistics by Stein's method', *arXiv preprint arXiv:1603.01889*, pp. 1-32.

Gavahane, M., Sequeira, D., Pandey, A. and Shetty, A. 'Anti-phishing using Hadoop-framework'. *International Conference on Technologies for Sustainable Development (ICTSD)*, Mumbai, India, 4-6 Feb. 2015, 1-4.

Gawade, N., Mundekar, S., Vare, N., Gada, R. and Bansod, S. (2018) 'URL Based Analysis for Phishing Detection Using Data Mining'.

Geng, G., Yan, Z., Chen, Y., Lee, X. and Li, X. 'Phishing detection based on newly registered domains'. *International Conference on Big Data (Big Data)*, Washington, DC, USA, 5-8 Dec. 2016: IEEE, 3685-3692.

Ghosh, A. (2013) 'Seclayer: A Plugin to Prevent Phishing Attacks', *The IUP Journal of Information Technology,* 9(4), pp. 52-64.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) 'Deep learning. Book in preparation for MIT Press', *URL¡ http://www. deeplearningbook. org*.

Google (2016) *Google safe browser*. Available at: https://developers.google.com/safe-browsing/?csw=1 (Accessed: 27 November 2016).

Government Communication Headquarter. (GCHQ, Centre, N.C.S. (2018) *Phishing attacks: defending your organisation.* London, United Kingdom: Centre for the Protection of National Security.

Gowtham, R. and Krishnamurthi, I. (2014) 'A comprehensive and efficacious architecture for detecting phishing web pages', *Computers Security,* 40, pp. 23-37.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R. and Schmidhuber, J. (2017) 'LSTM: A search space odyssey', *IEEE transactions on neural networks and learning systems,* 28(10), pp. 2222-2232.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. and Lew, M. S. (2016) 'Deep learning for visual understanding: A review', *Neurocomputing,* 187(2016), pp. 27-48.

Gupta, R. and Shukla, P. K. (2015) 'Performance Analysis of Anti-Phishing Tools and Study of Classification Data Mining Algorithms for a Novel Anti-Phishing System', *International Journal of Computer Network and Information Security (IJCNIS),* 7(12), pp. 70.

GüNeri, A. F., Ertay, T. and YüCel, A. (2011) 'An approach based on ANFIS input selection and modeling for supplier selection problem', *Expert Systems with Applications,* 38(12), pp. 14907-14917.

Hakkani-Tür, D., Tür, G., Celikyilmaz, A., Chen, Y.-N., Gao, J., Deng, L. and Wang, Y.-Y. 'Multi-Domain Joint Semantic Frame Parsing Using Bi-Directional RNN-LSTM'. *Interspeech*, San Francisco, USA, 8–12 September 2016, 715-719.

Han, W., Cao, Y., Bertino, E. and Yong, J. (2012) 'Using automated individual white-list to protect digital web identities', *Expert Systems with Applications,* 39(15), pp. 11861-11869.

Haruta, S., Asahina, H. and Sasase, I. 'Visual Similarity-Based Phishing Detection Scheme Using Image and CSS with Target Website Finder', *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Singapore, Singapore, 4-8 Dec. 2017: IEEE, 1-6.

Hassan, D. (2015) 'On Determining the Most Effective Subset of Features for Detecting Phishing Websites', *International Journal of Computer Applications,* 122(20).

Hawanna, V. R., Kulkarni, V. Y. and Rane, R. A. 'A novel algorithm to detect phishing URLs', *International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, Pune, India, 9-10 Sept. 2016: IEEE, 548-552.

Hong, J. (2012) 'The state of phishing attacks', *Communications of the ACM,* 55(1), pp. 74-81.

Hosoz, M., Ertunc, H. M. and Bulgurcu, H. (2011) 'An adaptive neuro-fuzzy inference system model for predicting the performance of a refrigeration system with a cooling tower', *Expert Systems with Applications,* 38(11), pp. 14148-14155.

Hu, J., Zhang, X., Ji, Y., Yan, H., Ding, L., Li, J. and Meng, H. 'Detecting Phishing Websites Based on the Study of the Financial Industry Webserver Logs', *3rd International Conference on Information Science and Control Engineering (ICISCE)*, Beijing, China, 8-10 July 2016: IEEE, 325-328.

Huang, H., Qian, L. and Wang, Y. (2012) 'An SVM- based technique to detect phishing URLs', *Journal of Information Technology,* 11(7), pp. 921-925.

Huh, J. H. and Kim, H. 'Phishing detection with popular search engines: Simple and effective'. *International Symposium on Foundations and Practice of Security*, Paris, France, 12-13 May 2011: Springer, 194-207.

Imani, M. and Montazer, G. A. (2017) 'Phishing website detection using weighted feature line embedding', *The ISC International Journal of Information Security,* 9(2), pp. 49-61.

Inuwa-Dutse, I., Liptrott, M. and Korkontzelos, I. (2018) 'Detection of spam-posting accounts on Twitter', *Neurocomputing,* 315(2018), pp. 496-511.

Islam, R. and Abawajy, J. (2013) 'A multi-tier phishing detection and filtering approach', *Journal of Network and Computer Applications,* 36(1), pp. 324-335.

Jain, A. K. and Gupta, B. (2017) 'Phishing Detection: Analysis of Visual Similarity Based Approaches', *Security and Communication Networks,* 2017.

Jain, A. K. and Gupta, B. B. (2016a) 'A novel approach to protect against phishing attacks at client side using auto-updated white-list', *Journal of Information Security (EURASIP),* 2016(1), pp. 1-11.

Jain, A. K. and Gupta, B. B. 'Comparative analysis of features based machine learning approaches for phishing detection'. *3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, 16-18 March 2016: IEEE, 2125-2130.

Jain, A. K. and Gupta, B. B. (2019) 'A machine learning based approach for phishing detection using hyperlinks information', *Journal of Ambient Intelligence and Humanized Computing,* 10(5), pp. 2015-2028.

Jansson, K. and von Solms, R. (2013) 'Phishing for phishing awareness', *Behaviour & information technology,* 32(6), pp. 584-593.

Jeeva, S. C. and Rajsingh, E. B. (2016) 'Intelligent phishing url detection using association rule mining', *Human-centric Computing and Information Sciences,* 6(1), pp. 1-19.

Kalola, P. V., Patel, S. and Pandit, R. 'PhishIdentifier: A Mozilla Firefox plugin to protect user against Phishing attacks'.

Karaboga, D. and Kaya, E. (2016) 'An adaptive and hybrid artificial bee colony algorithm (ABC) for ANFIS training', *Applied Soft Computing,* 49, pp. 423-436.

Kazemian, H. B. and Ahmed, S. (2015) 'Comparisons of machine learning techniques for detecting malicious web pages', *Expert Systems with Applications,* 42(3), pp. 1166-1177.

Khadir, R. (2015) 'Efforts and Methodologies used in Phishing Email Detection and Filtering: A Survey', *International Journal of Advanced Research in Computer Science,* 6(2).

Kumar, S., and Promma Phrommathed (2013) *Research Methodology.* US: Springer.

Kumar, V. and Kumar, R. 'Detection of a phishing attack using visual cryptography in ad-hoc network'. *International Conference on Communications and Signal Processing (ICCSP)*, Melmaruvathur, India, 2-4 April 2015: IEEE, 1021-1025.

Kumar, V. and Kumar, R. 'Detection of phishing attack using visual cryptography in ad hoc network'. *International Conference on Communications and Signal Processing (ICCSP),*: IEEE, 1021-1025.

Kuo, J. H., Lee, C. P. and Lee, J. K. (2018) *Phishing detection with machine learning*. Google Patents. [Online].

Le, A., Markopoulou, A. and Faloutsos, M. (2010) 'PhishDef: URL Names Say It All'.

Le, H., Pham, Q., Sahoo, D. and Hoi, S. C. 'URLnet: Learning a URL representation with deep learning for malicious URL detection', *arXiv preprint arXiv:1802.03162*, Washington, DC, US, 2 March 2018.

Lee, J.-L. and Park, D.-h. (2016) 'Phishing Detection Using Web Site Heuristics', *International Information Institute (Tokyo),* 19(2), pp. 523-530.

Li, H., Wang, S. and Kot, A. C. (2017) 'Image recapture detection with convolutional and recurrent neural networks', *Electronic Imaging,* 2017(7), pp. 87-91.

Li, J. and Wang, S. 'PhishBox: An Approach for Phishing Validation and Detection', *15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, Orlando, FL, USA, 6-10 Nov. 2017: IEEE, 557-564.

Li, L., Berki, E., Helenius, M. and Ovaska, S. (2014) 'Towards a contingency approach with whitelist-and blacklist-based anti-phishing applications: what do

usability tests indicate?', *Behaviour & Information Technology,* 33(11), pp. 1136-1147.

Li, Y., Yang, L. and Ding, J. (2016) 'A minimum enclosing ball-based support vector machine approach for detection of phishing websites', *Optik - International Journal for Light and Electron Optics,* 127(1), pp. 345-351.

Li, Y., Yang, Z., Chen, X., Yuan, H. and Liu, W. (2019) 'A stacking model using URL and HTML features for phishing webpage detection', *Future Generation Computer Systems,* 94(2019), pp. 27-39.

Lin, C., Tien, C., Chen, C. and Pao, H. 'Efficient spear-phishing threat detection using hypervisor monitor', *International Carnahan Conference on Security Technology (ICCST)*, Taipei, Taiwan, 21-24 Sept. 2015: IEEE, 299-303.

Liu, G., Qiu, B. and Wenyin, L. 'Automatic detection of phishing target from phishing web page'. *20th International Conference on Pattern Recognition (ICPR)*, Istanbul, Turkey, 23-26 Aug. 2010: IEEE, 4153-4156.

Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y. and Alsaadi, F. E. (2017) 'A survey of deep neural network architectures and their applications', *Neurocomputing,* 234, pp. 11-26.

M Jameel, N. and George, L. (2013) 'Detection of Phishing Emails using Feed Forward Neural Network', *International Journal of Computer Applications,* 77(7), pp. 10-15.

Machado, L. and Gadge, J. 'Phishing Sites Detection Based on C4.5 Decision Tree Algorithm', *International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, San Jose, CA, USA, 17-18 Aug. 2017: IEEE, 1-5.

Mao, J., Li, P., Li, K., Wei, T. and Liang, Z. 'BaitAlarm: detecting phishing sites using similarity in fundamental visual features'. *5th International Conference on Intelligent Networking and Collaborative Systems (INCoS), 2013*, Xi'an, China, 9-11 Sept. 2013: IEEE, 790-795.

Mao, J., Tian, W., Li, P., Wei, T. and Liang, Z. (2017) 'Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity', *IEEE Access,* 5, pp. 17020-17030.

Marchal, S., Armano, G., Gröndahl, T., Saari, K., Singh, N. and Asokan, N. (2017) 'Off-the-Hook: An Efficient and Usable Client-Side Phishing Prevention Application', *IEEE Transactions on Computers,* 66(10), pp. 1717-1733.

Marchal, S., François, J., State, R. and Engel, T. (2014) 'PhishStorm: Detecting Phishing With Streaming Analytics', *IEEE Transactions on Network and Service Management,* 11(4), pp. 458-471.

Marchal, S., Saari, K., Singh, N. and Asokan, N. 'Know Your Phish: Novel Techniques for Detecting Phishing Sites and Their Targets'. *36th International Conference on Distributed Computing Systems (ICDCS)*, Nara, Japan, 27-30 June 2016: IEEE, 323-333.

Marcolin, C. and Becker, J. (2016) 'Exploring Latent Semantic Analysis in a Big Data (base)'.

Mathieu, M., Henaff, M. and LeCun, Y. (2013) 'Fast training of convolutional networks through ffts', *arXiv preprint arXiv:1312.5851*, pp. 1-9.

Microsoft (2015) *Security in Internet Explorer*. Available at: https://www.microsoft.com/en-us/safety/pc-security/ie.aspx (Accessed: 29 November 2016).

Mishra, A. and Gupta, B. (2018) 'Intelligent phishing detection system using similarity matching algorithms', *International Journal of Information and Communication Technology,* 12(1-2), pp. 51-73.

Moghimi, M. and Varjani, A. Y. (2016) 'New rule-based phishing detection method', *Expert Systems with Applications,* 53(2016), pp. 231-242.

Mohammad, R. M., Thabtah, F. and McCluskey, L. 'An assessment of features related to phishing websites using an automated technique'. *International Conference on Internet Technology And Secured Transactions*, London, UK, 10-12 Dec. 2012: IEEE, 492-497.

Montavon, G., Samek, W. and Müller, K.-R. (2018) 'Methods for interpreting and understanding deep neural networks', *Digital Signal Processing,* 73(2018), pp. 1-15.

Moradpoor, N., Clavie, B. and Buchanan, B. 'Employing machine learning techniques for detection and classification of phishing emails', *Computing Conference*, London, UK, 18-20 July 2017: IEEE, 149-156.

Mouratidis, H., Kalloniatis, C., Islam, S., Huget, M.-P. and Gritzalis, S. (2012) 'Aligning Security and Privacy to Support the Development of Secure Information Systems', *Journal of Universal Computer Science,* 18(12), pp. 1608-1627.

Nanda, A. and Gupta, H. (2015) 'Anti- phishing techniques in cryptography', *International Journal of Electrical and Computer Engineering,,* 5(6), pp. 1511-1515.

National Cyber Security Centre (2018) *Software secure? What about your hardware?* [Weekly Threat Report], London, United Kingdom: National Cyber Security Centre. Available at: https://www.ncsc.gov.uk/report/weekly-threat-report-17th-august-2018 (Accessed: 15 September 2018).

Nguyen, L. A. T., Nguyen, H. K. and To, B. L. (2016) 'An efficient approach based on neuro-fuzzy for phishing detection', *Journal of Automation and Control Engineering,* 4(2), pp. 159-165.

Oest, A., Safei, Y., Doup, A., x00E, Ahn, G., Wardman, B. and Warner, G. 'Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis', *APWG Symposium on Electronic Crime Research (eCrime)*, San Diego, CA, USA, 15-17 May 2018: IEEE, 1-12.

Office for National Statistics (2017) *Crime in England and Wales: Year ending Dec. 2016*, London, United Kingdom: Office for National Statistics. Available at: https://www.ons.gov.uk/peoplepopulationandcommunity/crimeandjustice/bulletins/crimeinenglandandwales/yearendingjune2016#whats-happening-to-trends-in-fraud (Accessed: 15 June 2017).

Paliwal, S., Anand, D. and Khan, S. (2016) 'Application of Rule-based Fuzzy Inference System in Prediction of Internet Phishing', *International Journal of Computer Applications,* 148(14), pp. 30-37.

Park, A. J., Quadari, R. N. and Tsang, H. H. 'Phishing website detection framework through web scraping and data mining', *8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Vancouver, BC, Canada, 3-5 Oct. 2017: IEEE, 680-684.

Pathak, P. and Nanded, Y. M. (2016) 'A Dangerous Trend of Cybercrime: Ransomware Growing Challenge', *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* 5(2), pp. 371-373.

Patil, P., Rane, R. and Bhalekar, M. 'Detecting spam and phishing mails using SVM and obfuscation URL detection algorithm', *International Conference on Inventive Systems and Control (ICISC)*, Coimbatore, India, 19-20 Jan. 2017: IEEE, 1-4.

Peng, T., Harris, I. and Sawa, Y. 'Detecting Phishing Attacks Using Natural Language Processing and Machine Learning', *12th International Conference on Semantic Computing (ICSC)*, Laguna Hills, CA, USA, 31 Jan.-2 Feb. 2018: IEEE, 300-301.

Pham, C., Nguyen, L. A. T., Tran, N. H., Huh, E. and Hong, C. S. (2018) 'Phishing-Aware: A Neuro-Fuzzy Approach for Anti-Phishing on Fog Networks', *IEEE Transactions on Network and Service Management,* 15(3), pp. 1076-1089.

Powers, D. M. (2011) 'Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation', *Journal of Machine Learning Technologies,* 2(1), pp. 37-63.

Pradeepthi, K. V. and Kannan, A. 'Performance study of classification techniques for phishing URL detection', *Sixth International Conference on Advanced Computing (ICoAC)*, Chennai, India, 17-19 Dec. 2014: IEEE, 135-139.

Prakash, P., Kumar, M., Kompella, R. R. and Gupta, M. 'Phishnet: predictive blacklisting to detect phishing attacks'. *Proceedings of IEEE on INFOCOM*, San Diego, CA, USA, 14-19 March 2010: IEEE, 1-5.

Purkait, S. (2012) 'Phishing counter measures and their effectiveness – literature review', *Information Management & Computer Security,* 20(5), pp. 382-420.

Purkait, S. (2015) 'Examining the effectiveness of phishing filters against DNS based phishing attacks', *Information and Computer Security,* 23(3), pp. 333-346.

Purkait, S., Kumar De, S. and Suar, D. (2014) 'An empirical investigation of the factors that influence Internet user's ability to correctly identify a phishing website', *Information Management & Computer Security,* 22(3), pp. 194-234.

Ramanathan, V. and Wechsler, H. (2012) 'phishGILLNET—phishing detection methodology using probabilistic latent semantic analysis, AdaBoost, and co-training', *EURASIP J. on Info. Security,* 2012(1), pp. 1-22.

Ramesh, G., Gupta, J. and Gamya, P. G. (2017) 'Identification of phishing webpages and its target domains by analyzing the feign relationship', *Journal of Information Security and Applications,* 35, pp. 75-84.

Rao, R. S., Vaishnavi, T. and Pais, A. R. (2019) 'CatchPhish: detection of phishing websites by inspecting URLs', *Journal of Ambient Intelligence and Humanized Computing*.

Safer-Networking (2016) *FileAlyzer - Spybot Anti-malware and Antivirus*. Available at: https://www.safer-networking.org/products/filealyzer/ (Accessed: 04 August 2016).

Sahingoz, O. K., Buber, E., Demir, O. and Diri, B. (2019) 'Machine learning based phishing detection from URLs', *Expert Systems with Applications,* 117(2019), pp. 345-357.

Sahoo, P. K. 'Data mining a way to solve Phishing Attacks', *International Conference on Current Trends towards Converging Technologies (ICCTCT)*, Coimbatore, India, 1-3 March 2018: IEEE, 1-5.

Saif, D., El-Gokhy, S. M. and Sallam, E. (2018) 'Deep Belief Networks-based framework for malware detection in Android systems', *Alexandria Engineering Journal,* 57(4), pp. 4049-4057.

Sankhyan, R., Shetty, A., Dhanopia, L., Kaspale, C. and Dantal, G. (2018) 'PDS-Phishing Detection Systems', *Safety,* 5(04), pp. 2429-2431.

Shabut, A. M., Lwin, K. T. and Hossain, M. A. 'Cyber attacks, countermeasures, and protection schemes — A state of the art survey'. *10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA)*, Chengdu, China, 15-17 Dec. 2016: IEEE, 37-44.

Shahriar, H. and Zulkernine, M. (2012) 'Trustworthiness testing of phishing websites: A behavior model- based approach', *Future Generation Computer Systems,* 28(8), pp. 1258-1271.

Shaikh, A. N., Shabut, A. M. and Hossain, M. A. 'A literature review on phishing crime, prevention review and investigation of gaps'. *10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA)*, Chengdu, China, 15-17 Dec. 2016: IEEE, 9-15.

Sharma, H., Meenakshi, E. and Bhatia, S. K. 'A comparative analysis and awareness survey of phishing detection tools', *2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, Bangalore, India, 19-20 May 2017: IEEE, 1437-1442.

Shekokar, N. M., Shah, C., Mahajan, M. and Rachh, S. (2015) 'An ideal approach for detection and prevention of phishing attacks', *Procedia Computer Science,* 49(2015), pp. 82-91.

Shirazi, H., Bezawada, B. and Ray, I. 'Kn0w Thy Doma1n Name: Unbiased Phishing Detection Using Domain Name Based Features'. *Proceedings of the 23rd ACM on Symposium on Access Control Models and Technologies*, Indianapolis, Indiana, USA, 13-15 June 2018: ACM, 69-75.

Shirsat, S. D. 'Demonstrating Different Phishing Attacks Using Fuzzy Logic', *Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, Coimbatore, India, 20-21 April 2018: IEEE, 57-61.

Shyni, C. E., Sundar, A. D. and Ebby, G. S. E. 'Phishing Detection in Websites using Parse Tree Validation', *Recent Advances on Engineering, Technology and Computational Sciences (RAETCS)*, Allahabad, India, 6-8 Feb. 2018: IEEE, 1-4.

Silic, M. and Back, A. (2016) 'The dark side of social networking sites: Understanding phishing risks', *Computers in Human Behavior,* 60, pp. 35-43.

Singh, P., Jain, N. and Maini, A. 'Investigating the effect of feature selection and dimensionality reduction on phishing website classification problem', *1st International Conference on Next Generation Computing Technologies (NGCT)*, Dehradun, India, 4-5 Sept. 2015: IEEE, 388-393.

Smadi, S., Aslam, N. and Zhang, L. (2018) 'Detection of online phishing email using dynamic evolving neural network based on reinforcement learning', *Decision Support Systems,* 107(2018), pp. 88-102.

Sonowal, G. and Kuppusamy, K. (2017) 'PhiDMA–A phishing detection model with multi-filter approach', *Journal of King Saud University-Computer and Information Sciences,* 30(5), pp. 1-14.

Sonowal, G. and Kuppusamy, K. (2018) 'MMSPhiD: a phoneme based phishing verification model for persons with visual impairments', *Information & Computer Security,* 26(5), pp. 613-636.

Srinivasa Rao, R. and Pais, A. R. 'Detecting Phishing Websites using Automation of Human Behavior'. *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*, Abu Dhabi, United Arab Emirates, 02-02 April 2017: ACM, 33-42.

Steer, J. (2017) 'Defending against spear-phishing', *Computer Fraud & Security,* 2017(8), pp. 18-20.

Subasi, A., Molah, E., Almkallawi, F. and Chaudhery, T. J. 'Intelligent phishing website detection using random forest classifier'. *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, Ras Al Khaimah, United Arab Emirates: IEEE, 1-5.

Sundermeyer, M., Schlüter, R. and Ney, H. 'LSTM neural networks for language modeling'. *Thirteenth annual conference of the international speech communication association*, Portland, OR, USA, 9-13 September 2012: ISCA, 194-197.

Tahir, M. A. U. H., Asghar, S., Zafar, A. and Gillani, S. 'A Hybrid Model to Detect Phishing-Sites Using Supervised Learning Algorithms'. *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, NV, USA, 15-17 Dec. 2016: IEEE, 1126-1133.

Tan, C. L., Chiew, K. L. and Sze, S. N. 'Phishing website detection using URL-assisted brand name weighting system'. *International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS) 2014*, Kuching, Malaysia, 1-4 Dec. 2014: IEEE, 054-059.

Tan, C. L., Chiew, K. L., Wong, K. and Sze, S. N. (2016) 'PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder', *Decision Support Systems,* 88, pp. 18-27.

Thaker, M., Parikh, M., Shetty, P., Neogi, V. and Jaswal, S. 'Detecting Phishing Websites using Data Mining'. *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, Coimbatore, India, 29-31 March 2018: IEEE, 1876-1879.

Thakur, H. and Kaur, S. (2016) 'A Survey Paper On Phishing Detection', *International Journal of Advanced Research in Computer Science,* 7(4), pp. 64-68.

Thiyagarajan, P., Venkatesan, V. P. and Aghila, G. 'Anti-Phishing Technique using Automated Challenge Response method'. *International Conference on*

Communication and Computational Intelligence (INCOCCI), Erode, India, India, 27-29 Dec. 2010: IEEE, 585-590.

Tripathi, D., Nigam, B. and Edla, D. R. (2017) 'A Novel Web Fraud Detection Technique using Association Rule Mining', *Procedia Computer Science,* 115(2017), pp. 274-281.

Tyagi, I., Shad, J., Sharma, S., Gaur, S. and Kaur, G. 'A Novel Machine Learning Approach to Detect Phishing Websites', *5th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, 22-23 Feb. 2018: IEEE, 425-430.

Ulqinaku, E., Lain, D. and Capkun, S. '2FA-PP: 2nd factor phishing prevention', *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, Miami, Florida, 15-17 May 2019. 3323404: ACM, 60-70.

Upadhyaya, A. (2012) 'Design & development of a plug-in for a browser against phishing attacks', *International Journal of Emerging Technology & Advanced Eng.,* 2(3), pp. 105-111.

Vargas, J., Bahnsen, A. C., Villegas, S. and Ingevaldson, D. 'Knowing your enemies: leveraging data analysis to expose phishing patterns against a major US financial institution', *APWG Symposium on Electronic Crime Research (eCrime)*, Toronto, ON, Canada, 1-3 June 2016: IEEE, 1-10.

Vazhayil, A., Vinayakumar, R. and Soman, K. 'Comparative Study of the Detection of Malicious URLs Using Shallow and Deep Networks', *9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Bangalore, India, 10-12 July 2018: IEEE, 1-6.

Verma, R. and Rai, N. 'Phish-IDetector: Message-ID based automatic phishing detection'. *12th International Joint Conference on e-Business and Telecommunications (ICETE)*, Colmar, France, 20-22 July 2015: IEEE, 427-434.

Vidal, R., Ma, Y. and Sastry, S. (2016) *Generalized Principal Component Analysis. Interdisciplinary applied mathematics* New York, NY: Springer.

Virvilis, N., Tsalis, N., Mylonas, A. and Gritzalis, D. 'Mobile devices: A phisher's paradise'. *11th International Conference on Security and Cryptography (SECRYPT)*, Vienna, Austria, 28-30 Aug. 2014: IEEE, 1-9.

Wang, T., Kannan, K. N. and Ulmer, J. R. (2013) 'The Association Between the Disclosure and the Realization of Information Security Risk Factors', *Information Systems Research,* 24(2), pp. 201-218,494-495.

Wardman, B., Stallings, T., Warner, G. and Skjellum, A. 'High-performance content-based phishing attack detection'. *eCrime Researchers Summit (eCrime), 2011*, San Diego, CA, USA, 7-9 Nov. 2011: IEEE, 1-9.

Weiss, K. R. and Khoshgoftaar, T. M. 'Detection of Phishing Webpages Using Heterogeneous Transfer Learning', *3rd International Conference on Collaboration and Internet Computing (CIC)*, San Jose, CA, USA, 15-17 Oct. 2017: IEEE, 190-197.

Wen, S., Zhao, Z. and Yan, H. 'Detecting Malicious Websites in Depth through Analyzing Topics and Web-pages'. *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy*, Guiyang, China, 16 - 19 March 2018: ACM, 128-133.

Wenyin, L., Fang, N., Quan, X., Qiu, B. and Liu, G. (2010) 'Discovering phishing target based on semantic link network', *Future Generation Computer Systems,* 26(3), pp. 381-388.

Whittaker, C., Ryner, B. and Nazif, M. (2010) 'Large-Scale Automatic Classification of Phishing Pages', *NDSS,* 10(2010), pp. 2010-2023.

Wu, T., Liu, S., Zhang, J. and Xiang, Y. 'Twitter spam detection based on deep learning'. *Proceedings of the Australasian Computer Science Week Multiconference*, Geelong, Australia, January 30 - February 03, 2017: ACM, 3.

Xiang, G., Hong, J., Rose, C. and Cranor, L. (2011) 'CANTINA+: A Feature- Rich Machine Learning Framework for Detecting Phishing Web Sites', *ACM Transactions on Information and System Security,* 14(2).

Xinming, Y., Jing, L. and Jun, Z. 'Security and performance joint analysis method for authentication protocol based on CPN models'. *International Conference on Computer Application and System Modeling (ICCASM),*, 22-24 Oct. 2010, V7-505-V7-511.

Xu, Z., Li, S. and Deng, W. 'Learning temporal features using LSTM-CNN architecture for face anti-spoofing'. *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Kuala Lumpur, Malaysia, 3-6 November 2015: IEEE, 141-145.

Xu, Z., Wang, H. and Jajodia, S. 'Gemini: An Emergency Line of Defense against Phishing Attacks'. *33rd International Symposium on Reliable Distributed Systems*, Nara, Japan, 6-9 Oct. 2014: IEEE, 11-20.

Yang, C.-C., Tseng, S.-S., Lee, T.-J., Weng, J.-F. and Chen, K. 'Building an anti-phishing game to enhance network security literacy learning'. *IEEE 12th International Conference on Advanced Learning Technologies (ICALT), 2012*, Rome, Italy, 4-6 July 2012: IEEE, 121-123.

Yang, H., Lin, K. and Chen, C. (2018) 'Supervised Learning of Semantics-Preserving Hash via Deep Convolutional Neural Networks', *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 40(2), pp. 437-451.

Yang, P., Zhao, G. and Zeng, P. (2019) 'Phishing Website Detection Based on Multidimensional Features Driven by Deep Learning', *IEEE Access,* 7, pp. 15196-15209.

Yao, W., Ding, Y. and Li, X. 'Deep Learning for Phishing Detection'. *Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Melbourne, Australia, 11-13 Dec. 2018: IEEE, 645-650.

Yu, Y., Gong, Z., Zhong, P. and Shan, J. 'Unsupervised Representation Learning with Deep Convolutional Neural Network for Remote Sensing Images'. *International Conference on Image and Graphics*, Cham: Springer, 97-108.

Yuan, H., Yang, Z., Chen, X., Li, Y. and Liu, W. 'URL2Vec: URL Modeling with Character Embeddings for Fast and Accurate Phishing Website Detection'. *Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Melbourne, Australia, 11-13 Dec. 2018: IEEE, 265-272.

Zareapoor, M. and Seeja, K. (2015) 'Feature Extraction or Feature Selection for Text Classification: A Case Study on Phishing Email Detection', *International Journal of Information Engineering and Electronic Business,* 7(2), pp. 60-65.

Zeng, Z., Jiang, X. and Neapolitan, R. (2016) 'Discovering causal interactions using Bayesian network scoring and information gain', *BMC Bioinformatics,* 17(1), pp. 221-234.

Zhang, H., Liu, G., Chow, T. W. and Liu, W. (2011) 'Textual and visual content-based anti-phishing: a Bayesian approach', *IEEE Transactions on Neural Networks,* 22(10), pp. 1532-1546.

Zhang, J., Wu, C., Li, D., Jia, Z., Ouyang, X. and Xin, Y. (2012) 'An empirical analysis of the effectiveness of browser-based antiphishing solutions', *International Journal of Digital Content Technology and its Applications,* 6(7), pp. 216-224.

Zhang, N. and Yuan, Y. (2013) *Phishing detection using neural network.* [Online] Available at: http//cs229.stanford.edu/proj2012/ZhangYuan-PhishingDetectionUsingNeuralNetwork.pdf (Accessed: 23 April 2016).

Zhang, W., Jiang, Q., Chen, L. and Li, C. (2017) 'Two-stage ELM for phishing Web pages detection using hybrid features', *World Wide Web,* 20(4), pp. 797-813.

Zhao, R., John, S., Karas, S., Bussell, C., Roberts, J., Six, D., Gavett, B. and Yue, C. (2017) 'Design and evaluation of the highly insidious extreme phishing attacks', *Computers & Security,* 70(2017), pp. 634-647.

Zhou, L., Pan, S., Wang, J. and Vasilakos, A. V. (2017) 'Machine learning on big data: Opportunities and challenges', *Neurocomputing,* 237(2017), pp. 350-361.

Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L. and Xie, X. 'Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks'. *Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, Arizona, USA, 12–17 February 2016: AAAI, 3697-3703.

Zhuang, W., Jiang, Q. and Xiong, T. 'An intelligent anti-phishing strategy model for phishing website detection'. *32nd International Conference on Distributed Computing Systems Workshops*, Macau, China, 18-21 June 2012: IEEE, 51-56.

Zouina, M. and Outtaj, B. (2017) 'A novel lightweight URL phishing detection system using SVM and similarity index', *Human-centric Computing and Information Sciences,* 7(1), pp. 17.

Çakıt, E. and Karwowski, W. (2017) 'Predicting the occurrence of adverse events using an adaptive neuro-fuzzy inference system (ANFIS) approach with the help of ANFIS input selection', *Artificial Intelligence Review,* 48(2), pp. 139-155.

Appendix

Appendix A: Hybrid Features Table

Text Features Approach	
Search index	
Page ranking	This feature was used to check the importance of the web page by counting the number of quality links to a page to determine the relevance of the site on the Internet.
Google index	This feature was used to compare if the URL of the website included in the Google index matched the one submitted to google index.
Website traffic	This feature is used to measure the amount of data sent and received by a visitor to a website.
Statistical-report	This feature is also used for the usage of the website, such as the number of queries and the website availabilities. However, a new website may fail this check; some other features are used to ascertain the legitimacy of a website.
Security & encryption	
Existence of "https" in URL	This feature was used to check the existence of https in a URL
Long URL	This feature was used to check the length of the URL to determine if the original website has the correspondent URL.
Using the IP address	This feature was used to check the URL if it contains IP address as most phishers use this to deceive the unsuspected user.
Abnormal URL	This feature will check the URL against abnormality in the resources locator against the information stored in the WHOIS database for the legitimate website.
Abnormal request	This feature checks if there is a request from an external object within the web page such as image or video loaded from another domain.
Abnormal Anchor	This feature checks if their anchor element is like a tag <a> from an external link. This feature is treated as the request URL.
Web address bar	
Adding prefix or suffix	This feature is used to check if the dash symbol that is rarely used in a valid URL. Phishers tend to add suffix or prefix to separate by (-) to the domain name to made users feel that they are dealing with the legitimate web page. These are checked in the URL with our approach.
URL is having "@" symbol	This feature is used to check for the @ symbol in the URL as it leads the browser to ignore everything preceding the @ symbol
Using URL shortening services	This feature checks for considerably smaller URL length and still leads to the acquired web page. These are achieved by using https redirect on a domain name that is short.
Some links are pointing to a page	This feature checks the number of links that are pointing to the web page.
Using non-standard port	This feature is useful as it checks for validating if a service such as https is up or down. If all ports are open, phishers can run almost any service they want. As a result, user information is threatened.
Domain identity	

Age of the domain	This feature is used to extract the information from the WHOIS database and compare with information of a phishing site. Most phishing websites live for a short period.
DNS record	This feature was used to check the identity of the domain in the WHOIS database for the records. However, If the DNS record is not found or empty, the website is then classified as a phishing web page.
Domain registration length	This use of this feature is to check how the site is registered. Since phishing websites live for a short period, we believe that trustworthy domains are usually paid for several years in advance.
Sub-domain	This use of this feature is to check how the site is registered.
Source code Javascript	
Redirect using "//."	This feature was used to check the existence of // within the URL path, which means that the user will be redirected to another website.
Submitting information to an email	This feature was used to check if a website redirected user's information to a personal email, instead of a server to process.
https	This feature is used to check the existence of secure communication and if the issuer is trusted and how long, the certificate is issued.
Frame Features Approach	
Iframe Redirection	This feature is used to check the HTML tag used to display additional web pages in the current website. A phisher will take advantage of it by making the tag invisible without a frame border.
Disabling right-click	This feature is used to check if the right-click function is disabled using the JavaScript so that users cannot save or view the web page's source code.
Using a pop-up window	This feature is used to check if users were asked to submit their personal information through a pop-up window, which is unusual to find in a legitimate website.
Server form handler (SHF)	This feature is used to check if the domain name in server form handler is different from the domain name of the web page
Website forwarding	This feature is used to check how many times a website has a redirect, a legitimate site does one time, while phishing site repeats this more than four times.
The link in Script & Meta	This feature is used to check that the tag on the website is linked to the same domain of the web page.
Layout similarity	This feature is used to check the percentage of the layout similarity of the web page.
Style similarity	This feature is used to check the percentage of the style similarity of the web page.
Image Features Approach	
Favicon	This feature is used to check the icon associated with a particular web page and check if the icon is loaded from a domain other than that is shown in the address bar.
Image size	This feature is used to check the size of the images on the website
Alternative text	This feature is used to check with some level percentage if the alternative text is used on the website

Mouse over	This feature is used to check if JavaScript is used to show a fake URL in the status bar to users.
Login form	This feature is used to check if there is an obstructive login form on the website

Appendix B: Source Code from MATLAB AppDesign for the Validation Toolbar

We show some important code use to develop our model testing interface for deep learning. The application uses the interface to validate the phishing website in our system implementation.

```
classdef PhishDect < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

PhishDectPanel matlab.ui.container.Panel

Check matlab.ui.control.Button

Label matlab.ui.control.Label

output matlab.ui.control.Lamp

urlEditFieldLabel matlab.ui.control.Label

link matlab.ui.control.EditField

end

properties (Access = public)

input = 'link.Value';

end

methods (Access = private)

% Button pushed function: Check

function CheckPushed(app, event)

load phishfin;

app.input = app.link.Value;

app.input = lower(app.input);

documentsNew = tokenizedDocument(app.input,'DetectPatterns','hashtag');
```

```
enc = wordEncoding(documentsNew);
app.input1 = doc2sequence(enc,documentsNew,'Length',75);
app.output = phishfin(input1);
app.output.Label = classify(app.phishfin,app.input1);
[app.input string(app.output.Label)];
[app.output string(app.input1)];
app.output.color = app.output.label;
if (app.output.label == 'phishing')
app.output.colour = [1.00,0.00,0.00];
elseif(app.output.label == 'suspicious')
179
app.output.colour = [1.00,1.00,0.00];
else
app.output.colour = [0.00,1.00,0.00];
end
end
end
% App initialization and construction
methods (Access = private)
% Create UIFigure and components
function createComponents(app)
% Create UIFigure
app.UIFigure = uifigure;
app.UIFigure.Position = [100 100 186 344];
app.UIFigure.Name = 'UI Figure';
% Create PhishDectPanel
app.PhishDectPanel = uipanel(app.UIFigure);
app.PhishDectPanel.Title = 'PhishDect';
app.PhishDectPanel.Position = [1 1 185 344];
```

```
% Create Check
app.Check = uibutton(app.PhishDectPanel, 'push');
app.Check.ButtonPushedFcn = createCallbackFcn(app, @CheckPushed, true);
app.Check.Position = [65 200 67 22];
app.Check.Text = 'Check';
% Create Label
app.Label = uilabel(app.PhishDectPanel);
app.Label.HorizontalAlignment = 'right';
app.Label.Position = [65 159 25 22];
app.Label.Text = '';
% Create output
app.output = uilamp(app.PhishDectPanel);
app.output.Position = [112 161 20 20];
% Create urlEditFieldLabel
app.urlEditFieldLabel = uilabel(app.PhishDectPanel);
app.urlEditFieldLabel.HorizontalAlignment = 'right';
app.urlEditFieldLabel.Position = [7 233 25 22];
app.urlEditFieldLabel.Text = 'url';
% Create link
app.link = uieditfield(app.PhishDectPanel, 'text');
app.link.Position = [47 233 100 22];
180
end
end
methods (Access = public)
% Construct app
function app = PhishDect
% Create and configure components
createComponents(app)
```

```
% Register the app with App Designer
registerApp(app, app.UIFigure)
if nargout == 0
clear app
end
end
% Code that executes before app deletion
function delete(app)
% Delete UIFigure when app is deleted
delete(app.UIFigure)
end
end
end
end
```

Appendix C: Features Extraction Code

```
static void extractFeatures(String URL)
      {
          int[] featureVector = new int[15];
          //feature 1 URL has ip address
                    string domainName =
extractDomainName("C:\\Users\\ResearchPC\\Documents\\extract_domain.py",
URL);
          //Regex ip = new Regex(@"\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\b");
          //MatchCollection result = ip.Matches(domainName);
          Console.WriteLine(domainName);
          domainName.TrimEnd('\r', '\n');
          //IPAddress ipAddress;
          //bool result = IPAddress.TryParse(domainName.Trim(), out ipAddress);
          //Console.WriteLine(result);
          if (parse(domainName) == true)
```

```
            featureVector[0] = -1;
        else
            featureVector[0] = 1;
        Console.WriteLine(featureVector[0]);
        // Console.WriteLine(parse(domainName));
        //feature 2 Long URL
        if (URL.Length < 54)
            featureVector[1] = 1;
        else if (URL.Length >= 54 && URL.Length <= 75)
            featureVector[1] = 0;
        else
            featureVector[1] = -1;
        //feature 3 tinyURL
        //see later easy to do extract domains from hrefs in page http://bit.do/list-of-
url-shorteners.php
        //feature 4 @ Symbol
        if (URL.Contains("@"))
            featureVector[2] = -1;
        else
            featureVector[2] = 1;
        //feature 5 // after 7th position
        if (URL.LastIndexOf("//") > 7)
            featureVector[3] = -1;
        else
            featureVector[3] = 1;
  //feature 6 - in domain
        if (domainName.Contains('-'))
            featureVector[4] = -1;
        else
            featureVector[4] = 1;
```

```
        //feature 7 dots in domain part
        string getSubDomainDomain =
extractDomainName("C:\\Users\\ResearchPC\\Documents\\extract_domain1.py",
URL);
        getSubDomainDomain.Trim();
        string[] temparr = getSubDomainDomain.Trim().Split(' ');
        int dotsCount1 = 0;
        try
        {
            string subdomain = temparr[1];
            dotsCount1 = temparr[1].Split('.').Length - 1;
          }
        catch (Exception e) {
            Console.WriteLine("No subdomain found");
        }
        int dotsCount = domainName.Trim().Split('.').Length - 1 + dotsCount1;
        if (dotsCount == 1)
            featureVector[5] = 1;
        else if (dotsCount == 2)
            featureVector[5] = 0;
        else
            featureVector[5] = -1;
        //feature 8 use of https certificate issuer is ignored
        if (!URL.Substring(0, 6).Contains("https"))
            featureVector[6] = -1;
        else
            featureVector[6] = 1;
        //feature 9 //domain registration length
        whois.MyMethod(URL);
        string[] my_arr = new string[4];
```

```
        my_arr = whois.getArr();
        Console.WriteLine("Array: ");
        for(int i = 0; i < my_arr.Length; i++)
        {
            Console.WriteLine(my_arr[i]);
        }
        //System.IO.StreamReader reader = new
System.IO.StreamReader(@"C:\imp.txt");
        int i1 = 0;
        while (i1 < 3)
        {
            Console.WriteLine("I am here");
            string line = my_arr[i1];
            string[] arr = line.Split('-');
            if (i1 == 2)
            {
                //string[] arr1 = arr[1].Trim().Split('-');
                int year = Int32.Parse(arr[0].Trim());
                //int month = Int32.Parse(arr1[1]);
                int currentYear = 2016;
                //int currentMonth = 5;
                Console.WriteLine("Year: " + year);
                if (currentYear - year <= 1)
                    featureVector[7] = -1;
                else
                    featureVector[7] = 1;
             }
            i1++;
        }
```

```
        //feature 10 favicon.ico http://stackoverflow.com/questions/5119041/how-can-i-get-a-web-sites-favicon

        //feature 11 not feasible

        //12 https in domain part

        if (domainName.Contains("https") || getSubDomainDomain.Contains("https"))

            featureVector[8] = -1;

        else

            featureVector[8] = 1;

        //13 Request URL

        IWebDriver webDriver = new ChromeDriver("C:\\Users\\ResearchPC\\Selenium Drivers");


webDriver.Manage().Timeouts().SetPageLoadTimeout(TimeSpan.FromSeconds(180));


webDriver.Manage().Timeouts().ImplicitlyWait(TimeSpan.FromSeconds(180));

        webDriver.Navigate().GoToUrl(URL);

        Console.WriteLine("SRC:\n");

        int srcCount = 0;

        int legalSrc = 0;

        int illegalSrc = 0;

        double legalPercentage = 0.0;

        double illegalpercentage = 0.0;

        try

        {

            ReadOnlyCollection<IWebElement> links = webDriver.FindElements(By.XPath("//*[@src]"));

            //ReadOnlyCollection<IWebElement> links = webDriver.FindElements(By.Name("src"));

            foreach (IWebElement webElement in links)

            {

                string link = webElement.GetAttribute("src");
```

```
                string attributeURL = link;
                Console.WriteLine(attributeURL);
                string domain1 =
extractDomainName("C:\\Users\\ResearchPC\\Documents\\extract_domain.py",
attributeURL);
                if (!domain1.Contains(domainName))
                    illegalSrc++;
                else
                    legalSrc++;
                srcCount++;
            }
            //Please take a look at code below
            if (srcCount != 0)
            {
                illegalpercentage = 100 * illegalSrc / srcCount;
                if (illegalpercentage < 22.0)
                    featureVector[9] = 1;
                else if (illegalpercentage >= 22.0 && illegalpercentage <= 61.0)
                    featureVector[9] = 0;
                else
                    featureVector[9] = -1;
            }
            else
            {
                featureVector[9] = 1;
            }
        }
        catch (Exception exception) { Console.WriteLine(exception.Message); }
        int hrefCount = 0;
        int legalHref = 0;
```

```
        int illegalHref = 0;
        double legalPercentageH = 0.0;
        double illegalpercentageH = 0.0;
        Console.WriteLine("HREF:\n");
        try
        {
            ReadOnlyCollection<IWebElement> links =
webDriver.FindElements(By.XPath("//*[@href]"));
            //ReadOnlyCollection<IWebElement> links =
webDriver.FindElements(By.Name("src"));
            foreach (IWebElement webElement in links)
            {
                string link = webElement.GetAttribute("href");
                string attributeURL = link;
                Console.WriteLine(attributeURL);
                string domain1 =
extractDomainName("C:\\Users\\ResearchPC\\Documents\\extract_domain.py",
attributeURL);
                if (!domain1.Contains(domainName))
                    illegalHref++;
                else
                    legalHref++;
                hrefCount++;
            }
            if (hrefCount != 0)
            {
                illegalpercentageH = 100 * illegalHref / hrefCount;
                if (illegalpercentageH < 31.0)
                    featureVector[10] = 1;
                else if (illegalpercentageH >= 31.0 && illegalpercentageH <= 67.0)
                    featureVector[10] = 0;
```

```
                else
                    featureVector[10] = -1;
                }
                else
                {
                    featureVector[10] = 1;
                }
            }
            catch (Exception exception) { Console.WriteLine(exception.Message); }
            //string  [] data = whois.Instantiate(URL.Trim());
            Console.WriteLine("Illegal Src Percentage: " + illegalpercentage + " Illegal
src: "+ illegalSrc + " Total src: " + srcCount);
            Console.WriteLine("Illegal Href Percentage: " + illegalpercentageH + " Illegal
href: " + illegalHref + " Total href: " + hrefCount);
            //feature 12 meta link script
            //meta
            int metaCounter = 0;
            int metaIllegal=  0;
            try
            {
                ReadOnlyCollection<IWebElement> metaTags =
webDriver.FindElements(By.TagName("meta"));
                foreach (IWebElement metatag in metaTags)
                {
                    int startIndex = 0;
                    string content = metatag.GetAttribute("content");
                    if (content.Contains("http") || content.Contains("http"))
                    {
                        if (!content.Contains(domainName))
                            metaIllegal++;
                    }
```

```
                metaCounter++;
            }
        }
        catch (Exception e3)
        {
            Console.WriteLine(e3.Message);
        }
        //Link
        int linkCounter = 0;
        int illegalLink = 0;
        try
        {
            ReadOnlyCollection<IWebElement> linkTags =
webDriver.FindElements(By.TagName("link"));
            Console.WriteLine("In Link: ");
            foreach (IWebElement linktag in linkTags)
            {
           //int startIndex = 0;
                string contentURL = linktag.GetAttribute("href");
                Console.WriteLine(contentURL);
                string domainLink =
extractDomainName("C:\\Users\\ResearchPC\\Documents\\extract_domain.py",
contentURL);
                if (!domainLink.Contains(domainName))
                    illegalLink++;
                linkCounter++;
            }
        }
        catch (Exception e3)
        {
            Console.WriteLine(e3.Message);
```

```
        }
        int scriptCounter = 0;
        int illegalScript = 0;
        try
        {
            ReadOnlyCollection<IWebElement> scriptTags =
webDriver.FindElements(By.TagName("script"));
            Console.WriteLine("In Script");
            foreach (IWebElement scripttag in scriptTags)
            {
                //int startIndex = 0;
                string contentURL = null;
                contentURL = scripttag.GetAttribute("src");
                contentURL.Trim();
                if(contentURL != null)
                    Console.WriteLine("ContentURL: " + contentURL);
                string domainLink = "";
                if (contentURL != "" || contentURL!= null)
                {
                    domainLink =
extractDomainName("C:\\Users\\Rushikesh.Dharmadhik\\Documents\\extract_domai
n2.py", contentURL);
                    Console.WriteLine("DomainLink: " + domainLink);
                    if(domainLink.Trim() != "Exception")
                        if (!domainLink.Contains(domainName))
                            illegalScript++;
                }
                scriptCounter++;
            }
        }
        catch (Exception e3)
```

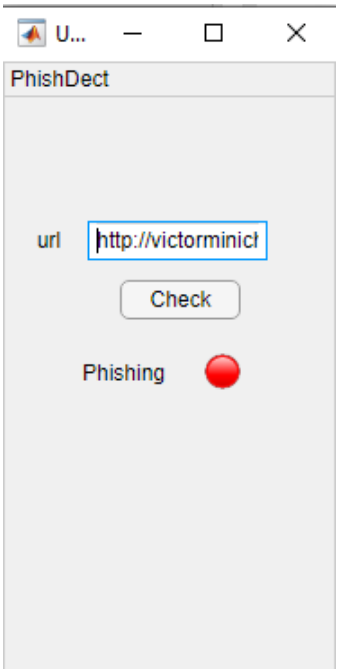
```
        {
            Console.WriteLine(e3.Message);
        }
        int totalLinkMetaScript = linkCounter + metaCounter + scriptCounter;
        int totalIllegal = illegalLink + metaIllegal + illegalScript;
        double illegalPercentageLinkMetaScript = 0.0;
        if (totalLinkMetaScript != 0)
            illegalPercentageLinkMetaScript = totalIllegal * 100 / totalLinkMetaScript;
        Console.WriteLine("Total Link, Meta, Script: " + totalLinkMetaScript + " Illegal:
" + totalIllegal + " "  + illegalPercentageLinkMetaScript);
        if (illegalPercentageLinkMetaScript < 17.0)
            featureVector[11] = 1;
        else if (illegalPercentageLinkMetaScript >= 17.0 &&
illegalPercentageLinkMetaScript <= 81.0)
            featureVector[11] = 0;
        else
            featureVector[11] = -1;
        try
        {
            ReadOnlyCollection<IWebElement> formTags =
webDriver.FindElements(By.TagName("form"));
            foreach(IWebElement formElement in formTags)
            {
                string action = formElement.GetAttribute("action");
                if (action.Contains("about:blank"))
                {
                    featureVector[12] = -1;
                    break;
                }
                else if (action.Contains("http") || action.Contains("https"))
                {
```

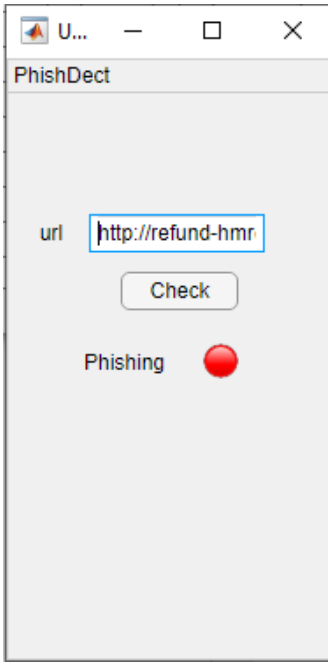
```
                if (!action.Contains(domainName))
                    featureVector[12] = 0;
                break;
            }
            else
                featureVector[12] = 1;
            }
        }
        catch(Exception e6) { Console.WriteLine(e6.Message);
            featureVector[12] = 1;
        }
        //Console.WriteLine("Data: ");
        //for (int i = 0; i < data.Length; i++)
        //  Console.WriteLine(data[i]);
        //feature 14 mail mailto doesnt seem promising
        //directly iFrame
        int counterIframes = 0;
        try
        {
            ReadOnlyCollection<IWebElement> iframeTags =
webDriver.FindElements(By.TagName("iframe"));
            foreach (IWebElement iframetag in iframeTags)
            {
                int startIndex = 0;
                string src = iframetag.GetAttribute("src");
                if (src.Contains("http") || src.Contains("https"))
                {
                    if (!src.Contains(domainName))
                        featureVector[13] = -1;
                    else
```

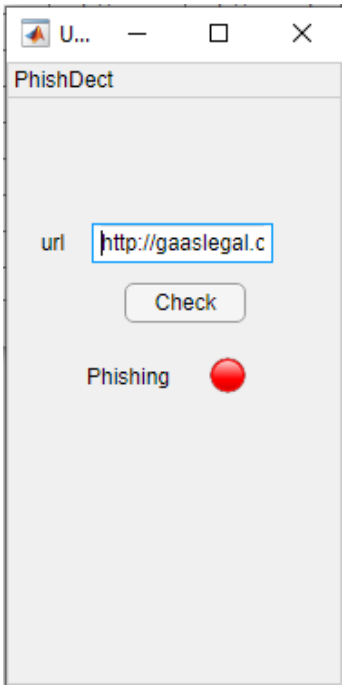
```
                featureVector[13] = 1;
            }
            else
                featureVector[13] = 1;
        }
    }
    catch (Exception exception5)
    {
        Console.WriteLine(exception5.Message);
        featureVector[13] = 1;
    }
    //feature age of domain
    string line1 = my_arr[0];
    string[] arr1 = line1.Split('-');
                //string[] arr1 = arr[1].Trim().Split('-');
        int year1 = Int32.Parse(arr1[0].Trim());
        int month = Int32.Parse(arr1[1].Trim());
        int currentYear1 = 2016;
        int currentMonth = 5;
    Console.WriteLine("In domain age: ");
        Console.WriteLine("Year: " + year1);
    Console.WriteLine("Month: " + month);
    int differenceyear = currentYear1 - year1;
    int differencemonth = currentMonth - month;
    int differencemonth1 = differenceyear * 12 + differencemonth;
    if (differencemonth1 >= 6)
        featureVector[14] = 1;
    else
        featureVector[14] = -1;
```

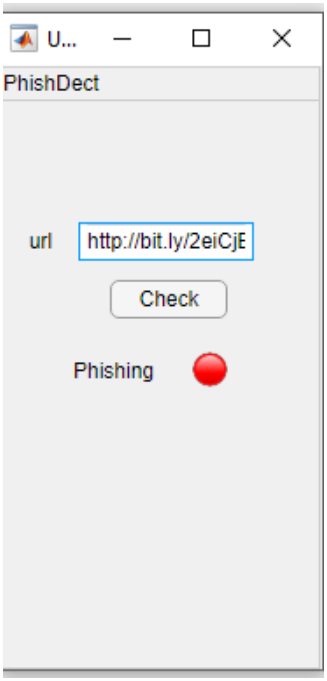
```
    Console.WriteLine("Features: ");
    for (int i = 0; i < featureVector.Length; i++)
        Console.WriteLine(featureVector[i]);
    //Console.WriteLine("FeatureVector:\n" + featureVector.ToString());
    Console.Read();
}
```

Appendix D: Phishing Website Validation

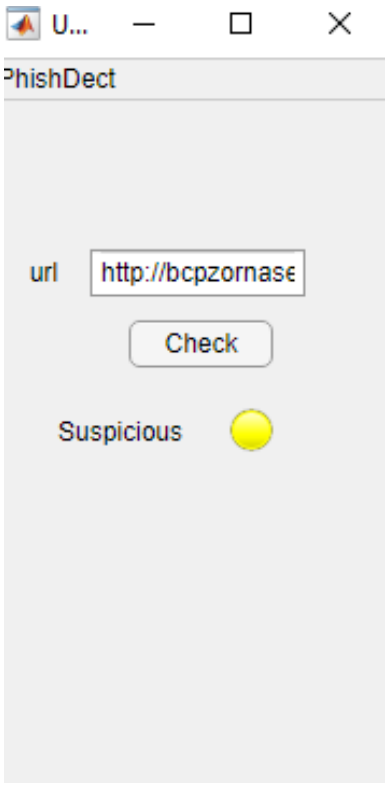


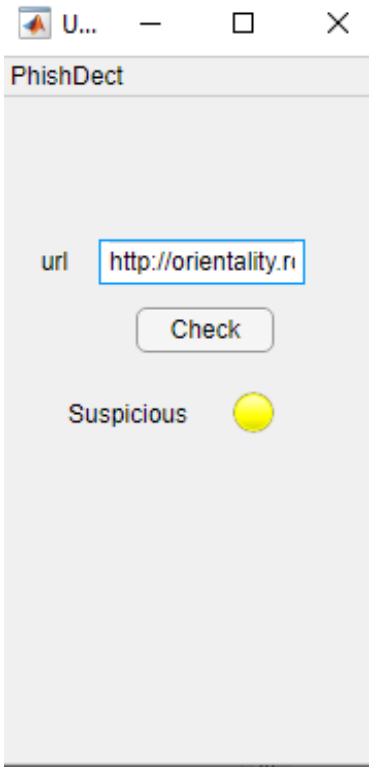


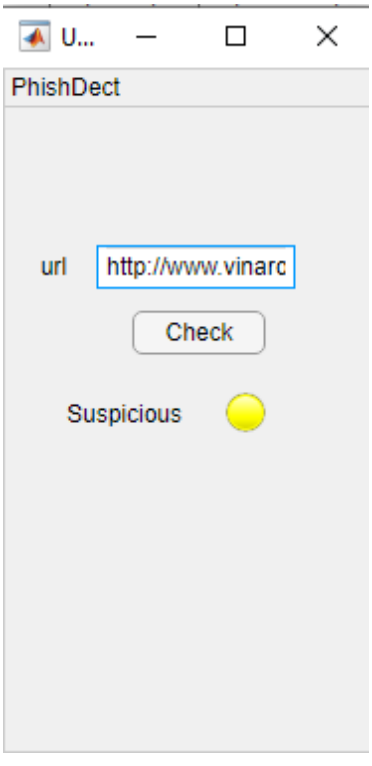


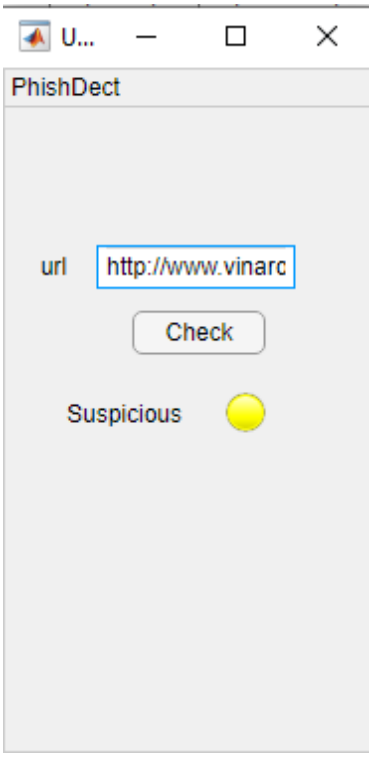


Appendix E: Suspicious Website Validation









Appendix F: Legitimate Website Validation