

An Efficient Phishing URLs Detection Approach Using Supervised Machine Learning



By

Muhammad Mustafa

Fall 2020 – MS (IS) - 328823

Supervisor

Dr. Mehdi Hussain

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of Masters
of Science in Information Security (MS IS)

In

School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),

Islamabad, Pakistan.

(August 2023)

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "An efficient phishing URLs detection approach using supervised machine learning " written by Muhammad Mustafa, (Registration No 00000328823), of SEECs has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____  _____

Name of Advisor: Dr. Mehdi Hussain _____

Date: 24-Jan-2023 _____

HoD/Associate Dean: _____

Date: _____

Signature (Dean/Principal): _____

Date: _____

Approval

It is certified that the contents and form of the thesis entitled "An efficient phishing URLs detection approach using supervised machine learning " submitted by Muhammad Mustafa have been found satisfactory for the requirement of the degree

Advisor : Dr. Mehdi Hussain

Signature:  _____

Date: 24-Jan-2023

Committee Member 1:Dr. Muhammad Zeeshan

Signature:  _____

26-Jan-2023

Committee Member 2:Dr. Hasan Tahir

Signature:  _____

Date: 24-Jan-2023

Signature: _____

Date: _____

Dedication

I extend my heartfelt gratitude to my parents, particularly my mother, whose unwavering support has been instrumental in my achievements. Additionally, I would like to express my sincere appreciation to my advisor, whose constant encouragement and guidance have been invaluable throughout this remarkable journey.

Certificate of Originality

I hereby declare that this submission titled "An efficient phishing URLs detection approach using supervised machine learning " is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name: Muhammad Mustafa

Student Signature: Mustafa

Acknowledgment

I begin by expressing my utmost gratitude to the Almighty Allah for granting me the fortitude and resilience to undertake this research endeavor. My sincere appreciation goes to Dr. Mehdi Hussain, my supervisor, whose unwavering guidance and mentorship have been invaluable throughout this journey. I am grateful to my colleagues and fellow researchers for their constant support and encouragement.

Lastly, I extend my heartfelt thanks to my parents for their unwavering prayers and unwavering support throughout this experience.

Table Of Contents

Thesis Acceptance Certificate.....	i
Approval Certificate	ii
Dedication	iii
Certificate Of Originality	iv
Acknowledgement	v
Table Of Contents	vi
List Of Tables	ix
List Of Figures	x
Abstract	xi
Introduction.....	1
1.1 Overview	1
1.2 Motivation.....	3
1.3 Research Objectives.....	4
1.4 Research Questions.....	4
1.5 Problem Statement.....	5
1.6 Solution Description.....	6
1.7 Contribution	6
1.8 Thesis Organization	7
1.9 Summary.....	8
Literature Review	9
2.1 Overview	9
2.2 User Education-Based Methodologies.....	10
2.3 Blacklisting-Based Methodologies.....	12
2.4 Visual Similarity-Based Methodologies.....	14
2.5 Machine Learning	15
2.5.1 Machine Learning Algorithms.....	15
2.5.2 Machine Learning-Based Schemes	20

2.6	Summary	23
	Research Methodology.....	24
3.1	Proposed Methodology	24
3.2	Design Objective	25
3.2.1	Malicious and Benign Websites' URLs Collection	26
3.2.2	Identification, Selection, and Filtering of Core Lexical Features	28
3.2.3	Phishing Detection by Employing Machine Learning Models	31
3.4	Summary	35
	Experimental Setup	36
4.1	Overview	36
4.2	Stages of the Experiment.....	36
4.3	Setting up Environment.....	37
4.4	Constructing the Dataset	37
4.5	Setting up the WEKA Tool	38
4.6	Creating Python Codebase.....	39
4.7	Installing Pre-requisite Softwares	40
4.8	Summary.....	41
	Experimental Results.....	42
5.1	Overview	42
5.2	Identification of Optimal Features Set.....	42
5.3	Evaluating Proposed Method.....	45
5.4	Summary.....	48
	Discussion and Analysis.....	49
6.1	Overview	49
6.2	Performance of Reference Method	51
6.3	Performance Comparison Of Proposed And Reference Methods	53
6.4	Applicability of the Approach	60
6.5	Summary.....	61
	Conclusion & Future Work	62
7.1	Conclusion	62

7.2	Limitation & Future Work	63
	Bibliography.....	64

List Of Tables

Table 1 A Brief Summary of Existing Machine Learning-based Approaches ..	22
Table 2 List of Already Extracted/Unfiltered Features	28
Table 3 Laptop Specifications	37
Table 4 Finalized Set of Features	43
Table 5 Proposed Scheme Results on All Nine Classifiers.....	46
Table 6 The Performance of Gupta et. al. Approach on KNN, Random Forest Logistic Regression and SVM classifiers.....	51
Table 7 Performance Evaluation of Gupta et. al. Results on Existing and New Dataset.....	52
Table 8 Performance Comparison of Proposed and Gupta et. al. Techniques on Larger 150k URLs Dataset	53
Table 9 Detection Results Using Proposed Methodology on Various Classification Machine Learning Algorithms	55
Table 10 Performance Comparison of Proposed and Existing Techniques	58

List Of Figures

Figure 1 Lexical Structure of a URL	3
Figure 2 Phishing Detection Analysis Types	9
Figure 3 Flow Diagram of Malicious and Benign Websites Classification	26
Figure 4 Representation of Dataset Collected and Stored in the CSV Format	27
Figure 5 Random Forest Classifier Model in WEKA	38
Figure 6 Implementation of Random Forest Python Code in Spyder IDE	39
Figure 7 Feature Importance of the Final Selected Set of Features	43
Figure 8 Comparison of Detection Accuracy of the Proposed and Benchmark Approach	54
Figure 9 Confusion Matrices of Employed ML Classifiers	56
Figure 10 ROC Curves of All Nine Algorithms Implemented	57
Figure 11 Comparison of No. of Dataset Entries and the Detection Accuracy of Proposed and Existing Approaches	57

Abstract

A phishing attack is an instance of social engineering in which the perpetrator deceives the user to gain access to sensitive information and/or personal data without authorization. This attack vector has become a prevalent problem in recent years and can result in substantial financial damage, as well as the potential risk of identity theft, data loss, and long-term damage to an organization's reputation. In prior efforts to counter this attack vector, researchers employed machine learning-based approaches which are based on lexical analysis of URLs and make use of datasets containing websites' URLs. However, these approaches are effective only on smaller no of dataset entries and are unable to detect new phishing URLs. This research has optimized an existing anti-phishing methodology to function on a larger dataset of phishing website URLs. To this end, a dataset of 150,000 URLs is collected for experimentation, and a set of optimized lexical features is incorporated. To obtain the optimal set of features, the feature significance scheme is then employed, using Random Forest Python code to reduce the number of lexical features from 70 to 15. For experiments, nine different machine learning classification algorithms, such as Random Forest, Support Vector Machine, and Logistic Regression, were used to assess the results. Precision, Recall, F1 Score, and Accuracy metrics were evaluated in comparison to the benchmark study. In experiments, it is observed that the proposed methodology obtained high detection accuracies as compared to the benchmark approach on a larger phishing dataset (150k), where the kNN classifier achieved the best detection accuracy of 99.98%.

Chapter 1

Introduction

In this chapter, we will discuss the basic concepts of the domain, significance, and history of research work. This chapter also puts forward the road map of the thesis and briefly highlights the further organization and structure of the thesis. The motivation for carrying out this research work has been explained in this chapter. It also highlights the prior contributions, benefits, scope of the work, and the research's main objectives.

1.1 Overview

In the last decade, the number of active websites has increased exponentially with the advancement of digital technology. The first website was publicly unveiled in the year 1991 [1]. The landmark figure of 1 billion websites was crossed in September 2014. Despite some fluctuations in the coming months, the number of websites reached 1 billion again in March of 2016. In the year 2016 alone, the total number of sites increased from 900 million to 1.7 billion. As of May 2023, there are more than 1.97 billion websites [1].

As a result of such a massive increase in the number of sites, the number of users, and the paradigm shift towards the digital world, cyber-crimes have also been on the rise in the recent past. A phishing attack is a famous type of cyber-attack in which an attacker attempts to steal the credentials and valuable user information by pretending as a legitimate user i.e. Social Engineering attack. The first-ever phishing attack dates back to the 1990s when a group of attackers used random numbers to create fake credit cards and stole the credentials of legitimate users with the help

CHAPTER 1. INTRODUCTION

of the American Online (AOL) web portal [2]. According to recent research, phishing attack ranks second among the cyber-attacks causing the most financial damage [3]. On average businesses and large enterprises lose \$4.65 million on average per phishing attack. This is an indication of the financial and reputational damage this type of cyber-attack can cause.

There have been a number of software developed in the recent past which can detect a phishing website based on a specific criterion and then blacklist those sites permanently to protect the users from a phishing scam. Spoof Guard, Netcraft Anti-phishing Toolbar, and Google Safe Browsing are some of the most famous ones. But the attackers have become intelligent and are aware of the tricks they can use to deceive these kinds of blacklisting software. That is why there is a need for an anti-phishing solution that can detect new malicious websites on the run time as well in order to be efficient.

Currently, there are a number of machine learning-based anti-phishing models proposed that can detect new and old phishing sites based on the contents of the URLs of websites [21][22]. A thorough analysis of the contents of the URL is known as a “Lexical Analysis of the URLs”. Some of these methodologies will be discussed in the next chapter.

The complete structure of a sample URL is shown in figure 1. A URL is made up of several parts or sub-categories. The extensive semantic analysis of these categories is called the lexical analysis of URLs. There are seven different parts of a URL named Protocol, Domain Name, Path, Parameter, Subdomain, Top-level domains, and query. The way browsers communicate is identified via the content in the protocol part of the URL. Some of the most common protocols are HTTPS (Hypertext Transfer Protocol Secure), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), IMAP (Internet Message Access Protocol), etc. A domain name represents a unique identifier for each website which helps in the identification

CHAPTER 1. INTRODUCTION

of a website. The path is the exact location on the web server where the file or directory is stored. A subdomain is a sub-category of the domain name. A domain name always consists of a top-level domain. A query is presented on dynamic web pages and is followed immediately by a question mark. It is also presented in the URL when the client makes a request to the server for a particular webpage based on a specific query.

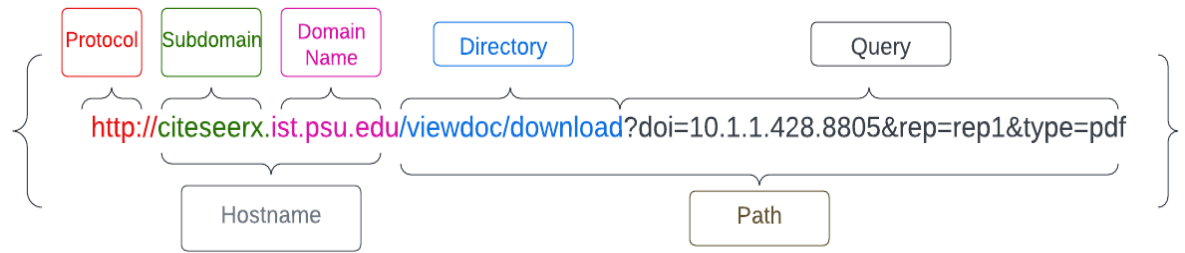


Figure 1: Lexical Structure of a URL

1.2 Thesis Motivation

The focus of this research is on the detection of phishing websites via machine learning models based on lexical analysis. Lexical analysis means the semantic analysis of the content present in the URLs of the websites. In the literature review section, various lexical analysis-based anti-phishing approaches have been discussed along with their pros and cons. The motivation of this research is to optimize the performance of the anti-phishing approach of Gupta et. al. [4] by reducing the number of features and increasing the size of the dataset used for classification.

This research is inspired by Gupta et. al. [4] which focuses on the lexical analysis of URLs to detect phishing websites. The benchmark study proposed by Gupta et. al. [4] consists of 9 distinct features, implementing three renowned machine learning algorithms. The aim of the proposed solution is to optimize the benchmark approach by using a significantly larger dataset and a reduced set of features.

1.3 Research Objectives

The purpose of research is to perform a comprehensive study to solve a particular problem using extensive scientific experiments. In this research, in-depth research has been done to propose an anti-phishing methodology. The following are the main objectives of this research:

- Highlighting the types of phishing attacks and analyzing their effects
- Comparison of various types of techniques for phishing detection
- Application of machine-learning-based anti-phishing models
- Optimization of the set of lexical features
- Evaluate the performance of anti-phishing models on larger dataset

1.4 Research Questions

This section describes the following research questions which are devised to perform this study:

a) Why is this research required?

The IT industry has evolved rapidly in the last 2 decades. The total number of websites as well as their users has increased exponentially. Similarly, website security attacks have also become common in the recent past. A Phishing attack is a type of social engineering attack, aimed at stealing the credentials of users by deceiving them. A Phishing website tricks the users into believing they are interacting with a legitimate website.

There are already various machine-learning-based approaches to classify benign and phishing websites to help the community. This research provides an extensive analysis of the prior methodologies in the same domain and suggests a light-weight anti-phishing approach.

b) What is the significance of the study? And what steps are involved in the research?

The purpose of the study is to classify phishing websites using a machine learning-based approach. It will help in the timely detection of phishing websites and will protect users from phishing attacks.

Our approach has been divided into the following broad steps in order to perform the research:

- Dataset collection
- Feature extraction
- Feature set optimization
- Generation of results
- Comparison of results
- Report writing

c) What are the aims of this study?

This research focuses on the following main aspects:

- a) Optimizing the results of the benchmark approach by increasing the length of the data set.
- b) Identifying the optimal set of features that can be used to generate high accuracies in the classification of phishing websites.

1.5 Problem Statement

A phishing attack is one of the most common cyber security attacks. In the first quarter of 2023, there were 562 million phishing emails detected worldwide. It is the highest number of monthly phishing attacks since the database started to track records. Overall, phishing attacks rank second among the cyber security attacks causing the most financial damage [3].

These numbers indicate that there is a need for an efficient phishing detection

approach that can timely detect phishing websites and protect users from reputational and financial damage. This study aims at proposing a light-weight anti-phishing solution that can classify phishing and benign websites by implementation of machine learning algorithms. The problem statement of this research is as follows:

“In literature, various phishing detection approaches exist, however, these methods are dependent on inefficient and large no. of lexical features. In addition, these methods are evaluated on a limited no. of URLs phishing datasets. Therefore, there is a need for an efficient and lightweight ML-based phishing detection model that requires less no. of features on the significantly larger dataset while achieving high accuracies.”

1.6 Solution Definition/Description

This study proposes an efficient light-weight phishing detection approach by implementing various machine learning algorithms. First, the data set was collected consisting of an equal number of benign and phishing URLs. It is pertinent to mention that the length of the data set used is significantly larger than the one used in the benchmark approach. Then, the lexical features of the URLs have been extracted and the redundant and insignificant features are removed. After the generation of the optimized set of features, nine machine-learning algorithms have been employed for the detection of phishing websites. The results have been obtained by the execution of python code and validated by WEKA tool.

1.7 Thesis Contribution

Our study “*An efficient phishing URLs detection approach using supervised machine learning*” successfully optimized the benchmark approach by obtaining almost similar detection accuracies while training and testing the model on a significantly larger dataset. Nine machine learning (ML) classifiers have been employed in this research which will be explained in Chapter 5. Following passage

CHAPTER 1. INTRODUCTION

illustrates the contributions of the proposed approach:

- Proposed method achieved similar phishing detection accuracy while enhancing the URL Phishing data set by 650%.
- The proposed scheme provides a lightweight URL phishing detection solution while employing the most significant set of 15 features.

1.8 Thesis Organization

The thesis is structured into multiple chapters, each designed to offer a complete comprehension of the conducted research. Here is a concise overview of each chapter's content:

Chapter 2: Literature Review - This chapter delves into the existing body of knowledge and research related to phishing detection solutions. It examines previous work done in the field, highlighting key findings, methodologies, and approaches used by researchers.

Chapter 3: Research Methodology - Here, the research methodology employed in the study is outlined. It explains the approach taken to optimize the existing methodology and the rationale behind it. The chapter provides details on the data collection process, feature identification, and the implementation of machine learning classifiers.

Chapter 4: Pre-Experimental Stage and Experiment Setup - This chapter focuses on the preliminary stage of the experiment and the setup of the experimental environment. It discusses the steps taken to ensure the reliability and validity of the data and the experimental conditions.

CHAPTER 1. INTRODUCTION

Chapter 5: Experimental Results - The results obtained from the experimentation phase are presented and analyzed in this chapter. It showcases the performance metrics, such as accuracy, precision, recall, and F1 score, achieved by the proposed methodology.

Chapter 6: Critical Analysis and Comparison - Here, a critical analysis of the results is conducted, examining the limitations and strengths of the proposed approach. The chapter also includes a comparison of the obtained results with the benchmark study to assess the effectiveness and improvement achieved.

Chapter 7: Conclusion and Future Directions - The final chapter concludes the research and summarizes the key findings. It highlights the contributions of the study and discusses potential directions for further research in this field.

By organizing the thesis into these chapters, the research is presented in a logical and structured manner, allowing readers to follow the progression of the study from literature review to conclusion.

1.9 Summary

This chapter gives an overview of phishing attacks. The objective as well as the scope of this study have been briefly explained while the problem statement has also been addressed. The next chapter will shed light on the literature review.

Chapter 2

Literature Review

This chapter offers an overview of the existing literature in the relevant field. The research performed in the URL phishing detection domain over the years has been discussed in detail.

2.1 Overview

In the recent past, the number of phishing attacks has increased significantly. The attackers have performed sophisticated social engineering attacks with the help of phishing websites, which look identical to a legitimate site to an internet user.

Over the years, numerous studies have been conducted to explore the identification of phishing websites. The technical aspects of anti-phishing solutions can be divided into four major classes: User-education based methodologies, Blacklisting-based methodologies, Visual similarity-based methodologies, and Machine Learning-based methodologies [5][6][7][21]. The following section will discuss in detail the relevant research performed in these categories.

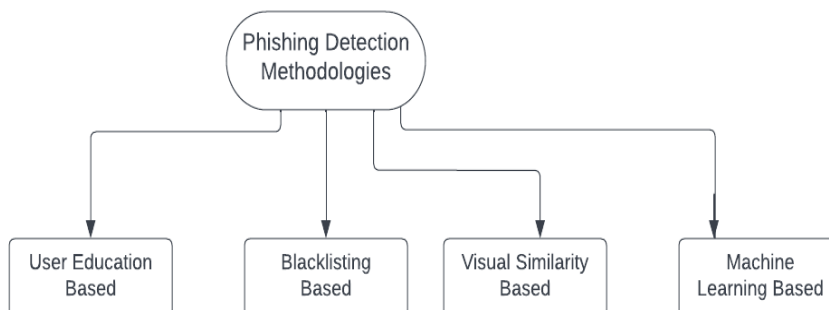


Figure 2: Phishing Detection Analysis Types

2.2 User Education-Based Methodologies

Currently, most of the internet users have limited information about cyber security threats. The attackers take advantage of this and deceive them via social engineering attacks. Phishing attacks, in particular, are based on deceiving the users, making them believe they are dealing with a legitimate entity. Hence, education and awareness related to phishing attacks for general internet users and employees in a security-sensitive organization are of paramount importance.

Nalin Asanka Gamagedara Arachchilage et. al. [5] researched focusing on the correlation between phishing threat awareness and the frequency of phishing attacks. The research revolved around 161 young computer users and considered their level of awareness of the phishing attack. This research proves via statistical analysis that phishing attacks can be avoided to a considerable extent if the users are well-trained and educated regarding such a massive threat. It also proved that those users with limited knowledge are a lot more susceptible to such social engineering attacks.

Similarly, Sadia Afroz et. al. [6] have also done an extensive analysis of the effects of computer users' education on the frequency of phishing attacks. The authors have developed a tool for detecting phishing websites by analyzing their content and structure. It uses a combination of machine learning algorithms and human feedback to identify and classify potential phishing sites. The tool allows users to submit suspected phishing websites for analysis and provides a real-time feed of the latest identified phishing sites. The authors concluded that only education and awareness are not sufficient and there should also be an efficient phishing detection mechanism in place, especially in security-sensitive companies.

Xun Dong et. al. [7] propose a new method for modelling the interaction between users and phishing attacks, with the aim of better understanding user behavior and identifying effective countermeasures. The authors develop a game-theoretic model that captures the strategic interaction between a phishing attacker and a user, and use simulations to study the impact of various factors on the success of phishing attacks.

The results show that user education can be effective in reducing the success of phishing attacks, but only when combined with technical countermeasures such as blacklists and phishing filters. The suggested model presents a valuable framework for exploring the intricacies of user-phishing interaction in future research endeavors.

Iacovos Kirlappos et. al. [8] have argued that traditional security education for phishing is insufficient and proposed a new approach to tackle this issue. The authors suggest that education should focus on the underlying cognitive mechanisms of phishing attacks and not just on the technical aspects. The paper highlights the role of context in phishing attacks and argues that education should provide users with tools to better evaluate context and make informed decisions. Additionally, the authors argue for the need to provide users with emotional and social support to better cope with phishing attacks. The paper concludes that a major rethink is needed to better educate users on phishing attacks and that a holistic approach that considers cognitive, emotional, and social factors is necessary.

Mohamed Alsharnouby et. al. [9] analyzed user strategies to combat phishing attacks and the reasons why phishing is still effective. The authors conducted semi-structured interviews with 22 participants who were targeted by phishing attacks. The results show that users have different strategies for identifying and avoiding phishing emails, including looking for specific cues, using their intuition, and relying on prior experience. However, these strategies can also lead to false positives and false negatives. Users also face cognitive limitations, such as decision fatigue and overconfidence, that can lead to errors in judgment. The authors suggest that future research should focus on developing effective training programs that consider the limitations and strategies of users. They also emphasize the need for a multi-layered defense approach that combines technical controls and user education to combat phishing attacks. The paper concludes by emphasizing the need for more research in this area and calls for a more holistic approach to combat phishing that incorporates technical, social, and psychological factors.

2.3 Blacklisting-Based Methodologies

Blacklisting phishing/malicious URLs is the more conventional or orthodox way of detecting phishing websites. In this method, a database is established and continuously updated to store malicious URLs, domains, and IP addresses. When a user visits a website, the URL is checked against the ones stored in the database. If the URL is among the blacklisted ones, access to that site is denied immediately and a phishing attack is prevented. If the URL is not found in the blacklist database, the website is declared benign, and the user is given permission to access that website.

The biggest disadvantage of black-list-based anti-phishing solutions, however, is that they cannot detect zero-day phishing attacks, i.e. a newly created phishing URL. If the URL of the spam website contains features that aren't already included in the blacklist, such a website will not be classified as a phishing one.

Srushti Patil et. al. [10] compared machine learning-based and blacklisting techniques for phishing website detection. The authors found that while blacklisting databases can be effective, they require regular updates to accurately detect new phishing websites. In fact, their results showed that the blacklisting methodology failed to detect a new website with a non-existent URL in the database. On the other hand, machine learning-based approaches can adapt to new phishing techniques and do not rely on database updates, making them more effective overall. However, these approaches may require more resources and expertise to implement and maintain. Ultimately, the study concludes that a hybrid approach that combines both techniques may be the best solution for detecting phishing websites.

Merlin. V. Kunju et. al. [11] compared various types of phishing detection techniques and highlighted the advantages and disadvantages of both blacklisting and machine learning-based approaches. They suggested that a hybrid solution combining the strengths of both approaches would be a more effective approach than relying solely on blacklist databases. The authors emphasized the importance of continuously improving and adapting phishing detection methods in order to keep up with evolving phishing tactics.

Routhu Srinivasa Rao et. al. [12] propose an enhanced blacklist-based approach for detecting phishing websites that analyzes their content and compares it to a pre-defined blacklist. The method employs feature extraction techniques to extract a set of characteristics from a website's content, such as URLs and images, and then evaluates them against the blacklist. Experimental results demonstrate that the proposed method can achieve a high level of accuracy in detecting phishing websites, outperforming existing blacklist-based methods. The authors concluded that the enhanced blacklist approach is a promising solution to detect phishing websites and improve the security of online transactions. The proposed approach can be used as a standalone solution or in conjunction with other techniques for effective and robust phishing detection.

Simon Bell et. al. [13] have evaluated the performance of three widely used phishing blacklists, namely Google Safe Browsing, OpenPhish, and PhishTank. The authors conducted an empirical study of the three blacklists to evaluate their accuracy, coverage, and timeliness in detecting phishing websites. They found that while Google Safe Browsing had the highest accuracy, it had the lowest coverage and was slow in adding new phishing URLs to its database. On the other hand, PhishTank had the highest coverage but the lowest accuracy. OpenPhish was found to strike a balance between accuracy and coverage. The authors concluded that using a combination of these blacklists could lead to better performance in phishing detection.

Steve Sheng et. al. [14] also presented an empirical analysis of popular phishing blacklists, including Google Safe Browsing, OpenPhish, and PhishTank. The authors tested these blacklists against a large dataset of phishing websites and found that they have varying levels of effectiveness in detecting and blocking such sites. The authors also identified some common limitations of these blacklists, such as false positives and false negatives. The paper concludes that while blacklists are an important tool for protecting users from phishing attacks, there is a need for ongoing improvements in their accuracy and coverage.

2.4 Visual Similarity-Based Methodologies

Phishing websites can also be identified through visual similarity approaches. In such techniques, a specific signature is used for the classification of phishing and benign websites. Phishing website signatures are made up of common features, such as malicious content and specific keywords, that are used to defraud users. To identify whether a website is phishing or not, that website is compared with the signature of a phishing website. On having a glance at the pages of the website, if its contents are similar to the phishing website, it is declared a fake website, and the user access is denied otherwise granted access to the specific URL.

Recently, Ankit et. al. [15] have analyzed the visual similarity-based anti-phishing approaches. They have highlighted the results of a survey, in which 90% of the participants failed to identify phishing sites based on their text content and visual appearance. The authors have stated that only a visual-based approach can be sometimes misleading and inefficient in terms of the identification of phishing websites.

Similarly, Saad Al-Ahmadi et. al. [16] applied the visual similarity technique alongside implementing the Convolution Neural Networks (CNN) algorithm while incorporating the URLs attributes. Here, the detection accuracy was achieved at around 99.67%.

Yu Zhou et. al. [17] have proposed a visual similarity-based anti-phishing approach that combines local and global image features to detect phishing websites. The method involves extracting image features using Scale-Invariant Feature Transform (SIFT) and Gabor filters, then comparing the features to those of legitimate websites. Experimental results show that the proposed approach can achieve high accuracy in detecting phishing websites and outperforms other state-of-the-art methods. The authors conclude that the combination of local and global features provides a robust and effective solution to the problem of detecting phishing websites.

2.5 Machine Learning

With the rise of sophisticated phishing attacks, researchers have turned to machine learning (ML) algorithms to effectively identify and categorize malicious websites. Decision tree, random forest, support vector machine, logistic regression, k-nearest neighbors, and neural networks are among the commonly employed ML algorithms for this task. Extensive research has demonstrated their ability to achieve impressive detection rates by leveraging suitable datasets, offering promising potential in mitigating the risks associated with phishing attacks.

2.5.1 Machine Learning Algorithms

This section explains a host of machine learning algorithms such as Random Forest, SVM, kNN, etc.

a) Random Forest:

The Random Forest algorithm is a widely recognized and robust machine learning technique employed for solving classification and regression tasks. By leveraging an ensemble approach, it combines multiple decision trees to build a comprehensive model that can effectively handle various types of datasets. [18]. The collective wisdom of each tree influences the classification outcome, with the majority of predictions from the decision trees determining the final decision. Random Forest is preferred over other algorithms because it provides good accuracy, handles missing data, and avoids overfitting. Additionally, it requires minimal parameter tuning and can handle high-dimensional data with ease. Random Forest also has a shorter training time in comparison to other algorithms, making it a popular algorithm for real-time phishing detection.

b) Logistic Regression:

Logistic Regression is a supervised learning algorithm that is widely used for binary classification problems. It calculates the probability of an input belonging to a particular class using a sigmoid function.

Based on the calculated probability, the input is then classified into one of two classes [19]. The algorithm works by finding the best-fit line or curve that separates the two classes in the dataset. Logistic Regression is known for its interpretability, robustness, and scalability, making it a popular choice in various domains. It can also be used for multiclass classification by extending the algorithm to include more than two classes.

c) KNN:

The k-Nearest Neighbors (kNN) algorithm is widely utilized in machine learning for tasks involving classification and regression. Its fundamental principle revolves around the assumption that points with similar attributes tend to be located close to each other in a graph or feature space [20]. In other words, neighboring points have a higher likelihood of belonging to the same class or exhibiting similar characteristics.

When working with kNN, the first step is to load the data containing various features and their corresponding class labels. The algorithm then measures the distances between the data points in the feature space. The most commonly used distance metric is Euclidean distance, although other distance measures can also be employed depending on the nature of the data.

Once the distances are calculated, kNN proceeds to classify a new or unlabeled data point by considering the k nearest neighbors. The user selects the value of k, which determines the number of neighbors to be taken into consideration. The new data point is assigned the class label that is most prevalent among its k nearest neighbors. For regression tasks, the algorithm can make predictions of continuous values by calculating the average of the target values from the k nearest neighbors.

It is worth emphasizing that selecting the right value for k is critical, as it can significantly influence the algorithm's performance. A small

value of k may result in overfitting, where noise or local irregularities in the data may influence the classification. On the other hand, a large value of k may introduce bias and oversimplification.

While kNN is a simple and intuitive algorithm, it does have some limitations. It can be computationally expensive, especially for large datasets, as it requires measuring distances between each data point. Additionally, the algorithm assumes that all features have equal importance, which may not always hold true. Feature scaling and weighting can be applied to address this issue.

kNN is a flexible and widely used algorithm that relies on the assumption that points in close proximity are likely to belong to the same class. By measuring distances between data points and considering the class labels of the k nearest neighbors, it enables classification or regression tasks. Careful selection of the value of k and addressing the computational complexity are essential considerations when employing the kNN algorithm in practice.

d) SVM:

Support Vector Machine (SVM) is a robust supervised machine learning algorithm widely recognized for its ability to tackle classification and regression problems. SVM shines in situations where data points can be effectively separated by boundary lines or hyperplanes.

The primary objective of Support Vector Machine is to identify the ideal hyperplanes that can effectively separate varying classes of data points [21]. This is achieved by maximizing the margin between the hyperplanes and the nearest data points of different classes. The support vectors, which are the data points nearest to the hyperplanes, play a vital role in determining the decision boundary.

SVM offers a notable benefit in its capability to handle datasets characterized by high-dimensional spaces. SVM can efficiently handle a large number of features, making it suitable for tasks involving

complex data. Furthermore, SVM can handle both linearly separable data, where a linear decision boundary suffices, and nonlinear data, by utilizing kernel functions that map the data into higher-dimensional spaces where linear separation becomes possible.

Another notable advantage of SVM is its capability to generalize well to new, unseen data points. This property is attributed to the concept of maximizing the margin, which encourages the algorithm to find a decision boundary that is less influenced by noisy or irrelevant data points. By focusing on the support vectors, SVM avoids overfitting and improves its ability to accurately predict unseen instances.

SVM finds applications in various fields and domains. In image classification, SVM has been utilized for tasks such as object recognition and face detection. In text classification, it has been employed for sentiment analysis, spam filtering, and document categorization. SVM has also found its place in bioinformatics, where it aids in tasks such as protein classification and gene expression analysis.

e) Naïve Bayes:

Naïve Bayes is a classification machine learning algorithm that is based on the famous Bayes theorem used for determining conditional probability. Based on the application of the Bayes theorem, this algorithm makes the prediction on the given dataset and the identified set of features [22]. It is widely considered as an extremely fast, reliable, and accurate algorithm for classification problems. There are three variations of Naïve Bayes that can be used accordingly: Multinomial Naïve Bayes Classifier, Bernoulli Naïve Bayes Classifier, and Gaussian Naïve Bayes Classifier.

f) Decision Tree:

The Decision Tree algorithm is widely employed in supervised machine learning for addressing both regression and classification

tasks. It makes predictions by learning specific decision rules depending on the dataset provided to the model. As the name indicates, there are several trees involved in the process of classification, that are sorted from root to leaves in a specific order. In the end, the nodes derived from the same parent are classified as the ones belonging to that particular class and the final classification process is achieved [23].

g) AdaBoost Classifier:

AdaBoost is a machine learning algorithm that is based on the principle of combining multiple weak classifiers to create a strong classifier. It is an ensemble-based method that iteratively combines several one-level decision trees to generate more accurate predictions. AdaBoost works by training multiple models sequentially and adjusting the weights of misclassified samples to ensure that they are correctly classified in the next iteration [24]. This approach improves the overall accuracy of the algorithm, making it a popular choice for solving classification problems. AdaBoost can also handle large datasets with multiple features, making it a versatile algorithm for machine learning tasks.

h) Gradient Boosting:

It is another example of an ensemble techniques-based algorithm that combines several weak classifiers to create a model that makes much more accurate classifications [25]. The difference between the two lies in the creation of so-called weak learners. While adaptive boosting focuses on modifying the weights attached to each instance, gradient boosting improves its predictions by retraining the model on the remaining errors of the strong learner. This process is called the gradient descent optimization process.

i) Extreme Gradient Boosting:

XGBoost, known as Extreme Gradient Boosting, is an enhanced iteration of the Gradient Boosting technique. It focuses primarily on improving the speed and the model's performance. In order to control the over-fitting,

XGBoost makes use of a much more regularized formation of the model, which improves its overall performance and speed [26]. In comparison, simple gradient boosting involves generic model formation, with no specific importance allocated to improved predictions and faster speed.

2.5.2 Machine Learning-Based Schemes

In the recent past, the implementation of machine learning algorithms for classifying benign and malicious websites has gained popularity. The biggest advantage of ML-based techniques is that a new spam website can also be detected [35]. The ML model's accuracy plays a crucial role in effectively detecting phishing attacks. Alswailem et. al. [27] proposed an anti-phishing solution based on the supervised machine learning algorithm Random Forest. The dataset under observation contains 6116 legitimate and phishing URLs. This research highlighted the 26 strongest features i.e. URL, Page content, Rank-based, etc. However, even though it obtained 98.8% of detection accuracy, the feature selection process could further be optimized due to its random selections. Huaping et. al. [28] also presented an anti-phishing solution that

is independent of any third-party influence. They employed both lexical as well as statistical 12 features based on websites using the ML model. The dataset of URLs consists of 6197 records only. The Deep Forest algorithm performed the best with 97.7% accuracy.

Orunsolu et. al. [29] have put forward an efficient ML-based phishing detection methodology. The evaluation of results has been conducted using Support Vector Machine (SVM) and Naïve Bayes (NB) classification algorithms. These algorithms have been applied to 15 lexical features extracted during the pre-processing stage. A dataset of 5041 entries has been divided into training and testing sets. On evaluation,

both SVM and NB yield 99.96% accuracy with a minimum response time. Although this approach provides high detection accuracies, the evaluation samples were limited. Vaibhav Patil et. al. [30] have also proposed a simple anti-phishing solution. For evaluating the results, machine learning classifiers like Random Forest, Logistic Regression, and Decision Tree have been implemented. The methodology has been executed on a dataset consisting of 6197 URLs with a focus on 12 lexical features. An impressive accuracy level of 96.58% has been attained.

Gururaj et. al. [31] have highlighted another machine-learning phishing detection approach. The dataset of 11056 URLs has been divided into training and testing sets. An accuracy of 96.87% has been yielded by implementing the Random Forest classifier. Alyssa Anne et. al. [32] have put forward another ML-based phishing-detection solution. Three renowned machine learning classifiers: kNN, Logistic Regression, and Decision Tree have been implemented in this approach. The dataset under observation consists of 5126 websites' URLs while 30 lexical features are considered. The highest detection accuracy obtained is 97.30% by Decision Tree. The approach used by Weiheng Bai et. al. [34] involves a dataset of 7058 URLs. To achieve optimal results, 12 Lexical features have been incorporated. Support Vector Machine is the classifier that produces the best accuracy of 95.15%. Selvakumari et. al. [35] have presented another anti-phishing solution based on lexical analysis of URLs. The dataset under observation consists of 95911 URLs of legitimate and phishing URLs. Five different ML classifiers have been executed out of which the Decision Tree algorithm yields the best accuracy of 95.50%.

One common limitation observed in all these approaches is the limited dataset size. It isn't unclear whether the proposed methodologies will yield similarly high accuracies once they are exposed to a significantly larger dataset. Moreover, some studies have incorporated random features to achieve maximum detection accuracies which means that the feature set is not the optimal one. Also, some of the proposed anti-phishing solutions are prone to search engine abuse.

Table 1: A Brief Summary of Existing Machine Learning-based Approaches

Papers	# of Features	# of Dataset entries	Classifier Algorithm	Best Accuracy Achieved	Limitations/ Future actions
Alswailem et. al. [27]	26	6116	Random Forest	98.8%	The feature selection process can be improved
Huaping Yuan et. al. [28]	12	6197	KNN, LR, RF, DT, GBDT, XGBST, DF	97.7%	Dataset too limited
A.A. Orunsolu et. al. [29]	15	5041	SVM NB	99.96%	Several features require fine-tuning,
Vaibhav Patil et. al. [30]	12	9076	RF LR DT	96.58%	Accuracy can be improved, no mention of response time
Gururaj et. al. [31]	30	11056	RF KNN DT SVM	96.87%	Dataset can be further processed to improve the accuracy
Alyssa Anne et. al. [32]	30	5126	KNN LR DT	97.30%	Dataset too limited
Suleiman Y. et. al. [33]	30	11055	RM NB SVM	97.30%	Fine-tuning required for some features
Weiheng Bai et. al. [34]	12	7058	LR NB DT SVM	95.15%	Feature set can be optimized
Selvaku mari et. al. [35]	12	95911	RF KNN LR DT	95.50%	Prone to search engine abuse

Although the current phishing detection techniques have shown remarkable results in terms of high detection accuracies on various ML models, there is still room for improvement. Most of the existing techniques rely on incorporating random lexical features to enhance the detection accuracy, which may lead to unnecessary

computational complexity. Additionally, the evaluation of these techniques has been conducted on a limited phishing dataset. To overcome these limitations, it is crucial to evaluate these methods on a larger phishing dataset using an efficient and relevant feature set. This will not only avoid unnecessary computational complexity but also improve the accuracy of the detection techniques. Therefore, in the proposed method, significant lexical features are employed for ML classification on a larger 150k URL dataset to achieve high detection accuracy. This approach will help in improving the reliability and effectiveness of phishing detection systems in real-world scenarios.

2.6 Summary

This chapter provided an overview of the background and existing research relevant to the thesis work. Various machine-learning algorithms have been briefly explained. The related literature has been elaborated alongside the critical analysis of the existing approaches. The limitations and the possible improvements are also identified. The following chapter throws light on the research methodology, followed during this thesis research.

Chapter 3

Research Methodology

In this chapter, the proposed methods will be briefly explained along with the main objective of this study. The process of dataset collection, and feature set construction has also been explained in this chapter.

3.1 Proposed Methodology

The following section presents a brief introduction to the research methodology employed in this study, aiming to optimize the approach proposed by Gupta et al. [4] while working with a considerably larger dataset. The primary objective is to achieve comparable accuracies to the benchmark approach. The proposed methodology builds upon the foundation laid by Gupta et al. while incorporating enhancements and modifications to accommodate the expanded dataset.

To accomplish this, a step-by-step analysis is conducted, taking into account the specific requirements and characteristics of the larger dataset. By leveraging appropriate techniques and algorithms, the methodology aims to address potential challenges and limitations encountered when working with an increased volume of data.

This research project aims to evaluate the efficacy of various machine learning classification algorithms in identifying and detecting phishing URLs. Nine classification algorithms, including Random Forest, Logistic Regression, kNN, and Support Vector Machine, were utilized to analyze and compare their performance. The study incorporated a dataset created by accessing various open-source databases such as PhishTank, OpenPhish, and Kaggle, which provided a diverse range of malicious and benign URLs.

CHAPTER 3. RESEARCH METHODOLOGY

The research consisted of several major stages to ensure comprehensive analysis and accurate results.

A. Collection of Malicious and Benign URLs: In this stage, a comprehensive collection of both malicious and benign website URLs was gathered from various sources. The heterogeneous dataset formed the basis for training and evaluating the machine learning models.

B. Identification, Selection, and Finalizing of Lexical Features: To develop an effective model, we identified and selected 15 core lexical features that were considered influential in distinguishing between malicious and benign URLs. These features were carefully analyzed, refined, and finalized for implementation.

C. Implementation of Machine Learning Classifiers: In this stage, the selected ML classifiers were implemented and trained using the dataset and the identified lexical features. The classifiers were fine-tuned and optimized to achieve accurate predictions of phishing URLs.

Following the aforementioned steps, the results (accuracy, precision, recall, f1 score) are obtained and are later used for further evaluations. The calculation of these evaluation metrics has been explained in the next section.

3.2 Design Objective

Figure 3 illustrates the architectural design of the proposed methodology. It depicts the flow of research from the dataset collection stage to the derivation of results and comparison with the benchmark approach. The key components in this diagram are further explained in the following passages.

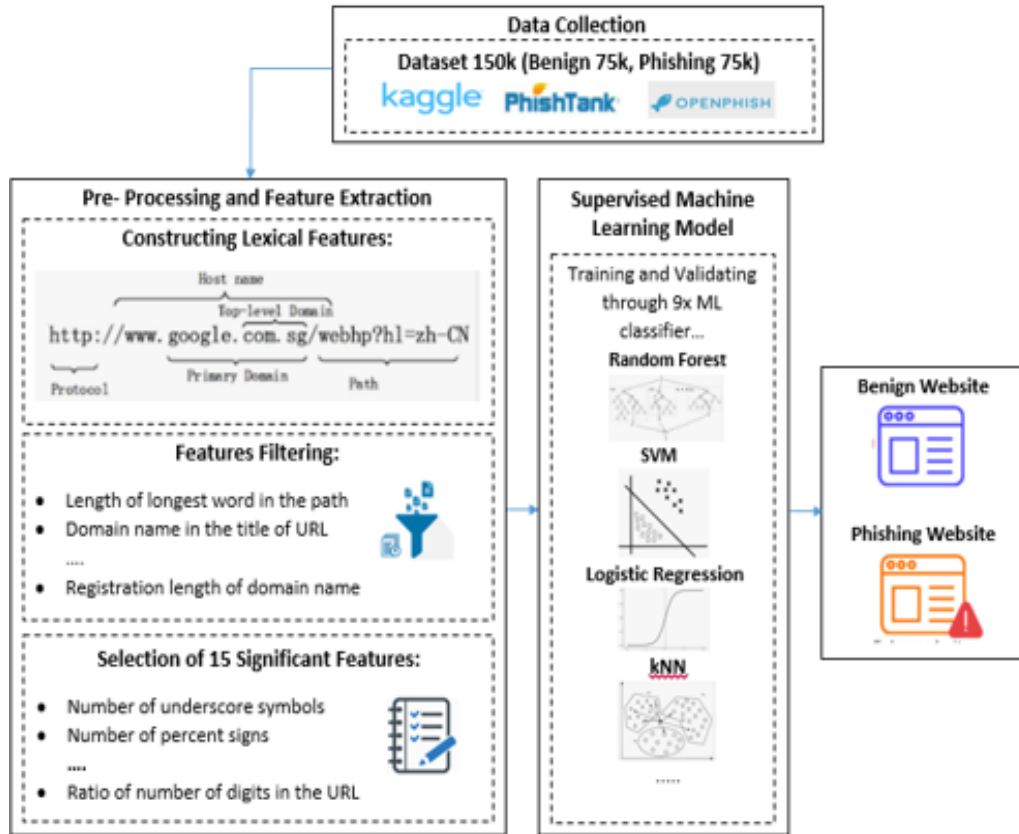


Figure 3: Flow Diagram of Malicious and Benign Websites Classification

3.2.1 Data Collection: Malicious and Benign Websites' URLs Collection

The importance of dataset collection before creating any machine learning model cannot be overstated. A well-curated and representative dataset serves as the foundation for accurate and reliable machine learning models. It ensures that the model learns from diverse and relevant examples, enabling it to make informed predictions and decisions in real-world scenarios. A comprehensive dataset helps in understanding the underlying patterns, relationships, and trends in the data, leading to more effective model training and evaluation. Additionally, a carefully collected dataset minimizes bias and ensures fairness, promoting ethical and responsible AI practices.

To conduct the research, a large number of URLs of both benign and phishing websites have been collected and analyzed. The initial dataset containing 20,000

CHAPTER 3. RESEARCH METHODOLOGY

samples was downloaded from the renowned database “Kaggle”. The dataset consisted of an almost identical number of legitimate and phishing websites.

In order to extend the aforementioned dataset, various other open-source sites have been accessed. Some of these websites include OpenPhish and PhishTank. A large number of the websites’ samples have been selected from these two famous databases. The goal was to compile a dataset that contains approximately 1,50,000 entries – significantly larger than the one used in the benchmark approach. In order to prevent data bias, it was ensured that almost half of the URLs (75000) belonged to phishing websites and the rest of the half (75000) were benign URLs.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	length_ur	length_ho	nb_dots	nb_hyph	nb_eq	nb_under	nb_perce	nb_comm	ratio_digi	longest_w	longest_w	longest_w	domain_i	domain_w	domain_r	status
2	46	20	3	0	1	0	0	0	0.108696	12	12	5	1	1	627	1
3	128	120	10	0	0	0	0	0	0.054688	35	35	0	1	0	300	1
4	52	25	3	0	0	0	0	0	0	17	17	9	1	0	119	1
5	21	13	2	0	0	0	0	0	0.142857	5	5	0	1	1	130	0
6	28	19	2	0	0	0	0	0	0	11	11	0	0	0	164	0
7	128	50	4	1	3	2	0	0	0.117188	17	13	17	1	1	25	1
8	50	15	2	0	1	0	0	0	0	9	7	9	1	0	705	1
9	51	14	5	0	0	0	0	0	0	7	7	7	1	0	0	1
10	35	22	2	0	0	0	0	0	0	14	14	4	0	1	67	0
11	22	15	2	0	0	0	0	0	0	8	8	0	1	1	148	0
12	57	14	3	1	0	0	0	0	0	8	7	8	1	0	496	1
13	39	30	2	2	0	0	0	0	0	9	9	0	1	1	0	0
14	44	18	3	0	0	0	0	0	0	11	11	7	1	0	329	1
15	28	21	2	0	0	0	0	0	0.035714	8	8	0	1	0	374	1
16	56	15	2	0	0	0	0	0	0.071429	24	7	24	1	1	204	0
17	110	30	2	1	1	0	0	0	0.1	46	12	46	1	0	381	1
18	40	17	3	0	0	0	0	0	0	9	9	4	0	0	617	0
19	62	16	2	2	0	0	0	0	0.080645	9	8	9	1	1	382	0
20	62	17	2	5	0	0	0	0	0.016129	9	9	7	1	0	346	0
21	69	20	2	1	0	0	0	0	0	15	12	15	1	1	360	1

Figure 4: Representation of Dataset collected and stored in the csv format

3.2.2 Identification, Selection, and Filtering of the Core Lexical Features

- A) *Importance of Feature Selection*: The collection of relevant features is the first step of a classification model. To obtain the best possible results, selecting the optimal features is of paramount importance.
- B) *Feature Extraction*: The initial dataset contained 20,000 sample URLs. It also contained 70 extracted features - such as the number of dots, domain registration length, number of equal signs, etc. - identifying the relevant lexical structure of each URL. Keeping these features as the basis, relevant lexical features were extracted from the rest of the URLs present in the extended dataset. In table 2, the extracted features in the initial dataset have been presented.

Table 2: List of Already Extracted/Unfiltered Features

Sr No	Feature
1	url
2	length_url
3	length_hostname
4	ip
5	nb_dots
6	nb_hyphens
7	nb_at
8	nb_qm
9	nb_and
10	nb_or
11	nb_eq
12	nb_underscore

CHAPTER 3. RESEARCH METHODOLOGY

13	nb_tilde
14	nb_percent
15	nb_slash
16	nb_star
17	nb_colon
18	nb_comma
19	nb_semicolumn
20	nb_dollar
21	nb_space
22	nb_www
23	nb_com
24	nb_dslash
25	http_in_path
26	https_token
27	ratio_digits_url
28	ratio_digits_host
29	punycode
30	port
31	tld_in_path
32	tld_in_subdomain
33	abnormal_subdomain
34	nb_subdomains
35	prefix_suffix
36	random_domain
37	shortening_service
38	path_extension
39	nb_redirection

CHAPTER 3. RESEARCH METHODOLOGY

40	nb_external_redirection
41	length_words_raw
42	char_repeat
43	shortest_words_raw
44	shortest_word_path
45	longest_word_host
46	longest_word_path
47	avg_words_raw
48	avg_word_host
49	avg_word_path
50	phish_hints
51	domain_in_brand
52	brand_in_subdomain
53	brand_in_path
54	suspicious_tld
55	statistical_report
56	nb_hyperlinks
57	ratio_intHyperlinks
58	ratio_extHyperlinks
59	ratio_nullHyperlinks
60	ratio_intRedirection
61	ratio_extRedirection
62	ratio_intErrors
63	login_form
64	external_favicon
65	links_in_tags
66	submit_email

CHAPTER 3. RESEARCH METHODOLOGY

67	ratio_intMedia
68	ratio_extMedia
69	page_rank
70	status

C) *Feature Ranking*: After extracting 70 lexical features from all the samples present in the dataset, the most significant or key features were selected in order to distinguish phishing sites from legitimate ones. Python code was used for ranking these features. The dataset stored in the form of the CSV file is read by python's built-in "pandas" library, and then it finds out the importance of each feature, arranging them in the descending order of significance. After identification of the importance associated with each feature, those features with negligible or extremely low values of importance were identified and removed from the features list. After execution of the Python code in several iterations, the 15 most significant features were selected, by removing the features with importance values lower than 0.01. In this way, through extensive evaluation, the features with extremely low significance were filtered out.

3.2.3 Employing of Machine Learning Models

A) *Division of dataset into training and testing*: After constructing the dataset and identifying the feature set, the subsequent stage entails applying machine learning classifiers to classify phishing and benign websites. In this study, the dataset comprises 150,000 URLs, with 80% of the samples allocated for training purposes. This ensures an ample training set for the machine learning algorithms to learn patterns and make accurate predictions. The remaining 20% of samples are reserved for evaluation, enabling the assessment of the models' performance and facilitating a comparison of the results.

B) *Python Code*: Python's "pandas" library has a pivotal role in the implementation of the proposed methodology. It provides a

versatile and efficient framework for data manipulation and analysis. The library's functionalities allow for seamless preprocessing of the dataset, handling missing values, and ensuring data integrity. Moreover, pandas enables efficient feature engineering, allowing the creation of informative and relevant features that contribute to the effectiveness of machine learning models.

- C) *Leveraging WEKA Tool:* The renowned WEKA (Waikato Environment for Knowledge Analysis) software is employed to evaluate the performance of the machine learning models and validate the results. WEKA offers a comprehensive suite of machine learning algorithms and tools, facilitating the creation, training, and evaluation of models. By leveraging the integration of Python and WEKA, the proposed methodology benefits from the strengths of both platforms. Python's flexibility and extensive libraries, combined with WEKA's robust algorithms and analysis capabilities, enhance the reliability and accuracy of the models.
- D) *Creation of ML Models:* In the implementation phase, nine different machine learning models are created using Python and WEKA. These models employ various classification algorithms, such as Random Forest, Logistic Regression, k-Nearest Neighbors (kNN), and Support Vector Machine (SVM). Each algorithm brings its unique characteristics and strengths, enabling a diverse ensemble of models to achieve robust predictions.
- E) *Results Generation:* Once the machine learning models are trained and evaluated, the performance metrics are computed. Evaluation metrics such as accuracy, precision, recall, and F1 score are used to assess the models' effectiveness in distinguishing between phishing and benign websites. These metrics provide valuable insights into the models' predictive power, highlighting their ability to accurately classify unseen data.

In the subsequent chapters, the evaluation of performance will be thoroughly discussed. The results obtained from each machine learning model will be analyzed, compared, and contrasted with the benchmark approach. Additionally, the implications and limitations of the proposed methodology will be explored, shedding light on potential areas for future research and improvement.

By leveraging the power of Python's code implementation and the integration with WEKA, this study presents a robust and comprehensive approach to phishing URL detection. The combination of diverse machine learning algorithms, extensive feature engineering, and thorough evaluation ensures the reliability and efficacy of the proposed methodology.

3.3 Evaluation Metrics

In the evaluation process, the following metrics have been taken into account: Precision, Recall, F1 score, and Accuracy. The following formulas explain and represent their respective definitions:

$$1) \text{ Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

$$2) \text{ Recall} = \frac{TP}{TP+FN}$$

$$3) \text{ Precision} = \frac{TP}{TP+FP}$$

$$4) \text{ F1 Score} = \frac{TP}{TP+\frac{1}{2}(FP+FN)}$$

In evaluating the performance of machine learning models, various metrics are employed to measure their effectiveness in classifying positive and negative instances. The calculation of these metrics involves different components, such as true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) [25].

True positives represent the count of correctly classified positive samples by the model. False positives refer to the misclassification of negative samples as positive. True negatives indicate the accurate identification of negative samples, while false negatives represent the misclassification of positive instances as negative [25].

To assess the model's ability to correctly identify positive instances, the recall metric is employed. It is calculated by dividing the total number of true positive instances by the sum of true positives and false negatives. Recall provides insights into the model's sensitivity to positive instances and its ability to avoid false negatives.

Precision, on the other hand, focuses on the accuracy of positive predictions. It is determined by dividing the number of true positive instances by the sum of true positives and false positives. Precision indicates how well the model avoids false positives and provides an indication of its specificity.

The F1 score is a metric that considers both precision and recall, offering a balanced measure of a model's performance. It is calculated as the harmonic mean of precision and recall, making it valuable for evaluating imbalanced datasets with unequal positive and negative instances.

In addition to recall, precision, and the F1 score, another important metric is accuracy. Accuracy represents the overall percentage of correctly classified instances, considering both true positives and true negatives. It provides a general measure of the model's correctness in predicting the class labels.

By considering these metrics, researchers can gain a comprehensive understanding of the model's performance and make informed decisions regarding its applicability in real-world scenarios. These metrics provide valuable insights into the model's strengths and weaknesses, allowing for further improvements and fine-tuning to enhance its performance.

3.4 Summary

This chapter briefly explained the proposed methodology and the various steps followed in order to perform the experiments. The design objective of this research has also been briefly discussed. The overall structure of the research i.e., collection of the dataset samples, selection of most significant features, and implementation of the machine learning classifiers have also been highlighted. The upcoming chapter will delve into the details of the experimental setup.

Chapter 4

Experimental Setup

In this chapter, we will highlight the experimental setup designed to perform the experiments for this study. The particular system configurations alongside the processes followed in this research have been explained.

4.1 Overview

The setup used for experimentations includes an extensive dataset of benign and malicious websites' URLs and a Laptop capable of running the WEKA tool and a Python codebase to create the machine learning model to obtain the results.

4.2 Stages Of The Experiment

For conducting the study, the first stage was to collect the relevant websites URLs to construct the database. As mentioned in Section 3.2, various open-source databases were searched for the construction of the final database containing 150k URLs. Next step was extracting the lexical features and identifying the optimal set of features. Subsequently, a range of machine learning classification algorithms were utilized to derive the outcomes.

The ML models were created using Python and the results were verified by the WEKA tool. The following passage will briefly explain the construction of the dataset, creation of the Python codebase and setting up of the WEKA tool.

4.3 Setting up Environment

A windows-based laptop has been used for carrying out the relevant experiments. The following table shows the specifications of this laptop:

Table 3: Laptop Specifications

Property	Description
<i>Manufacturer</i>	HP
<i>Model</i>	EliteBook 840 G3
<i>Operating System</i>	Windows 10 Pro
<i>Architecture</i>	x64 based
<i>Processor</i>	Intel Core i7 – 6600U @2.60 GHz, 6 th Generation
<i>RAM</i>	16 GB
<i>Storage</i>	1.5 TB

4.4 Constructing the Dataset

For constructing the final dataset, samples of benign and malicious URLs have been collected from various sites. The process of constructing the dataset was divided into two distinct phases. In the first step, the initial dataset containing 20,000 benign as well as phishing websites’ URLs were obtained from the renowned database “Kaggle”. In the next step, the dataset was extended by collecting the websites’ URLs from open-source databases such as PhishTank, OpenPhish, etc. Once these URLs were collected, the relevant lexical features were extracted from them by keeping the set of features of the initial dataset as the reference. The goal was to achieve a dataset that would be significantly larger than the one used by Gupta et al. The final dataset contains 150k benign and phishing websites’ URLs [36].

Chapter 5 will briefly explain the results obtained after performing the experiments.

4.5 Setting up the WEKA Tool

Waikato Environment for Knowledge Analysis, or simply WEKA is a famous software used for the implementation of various common machine learning algorithms. It provides an extremely simple user interface as well as a handful of useful options such as dataset preprocessing. It allows easy access to a number of classification and clustering-based machine learning algorithms [24].

For this research, the results have been deduced by first implementing the ML algorithms via WEKA and then later compared with the results obtained via the Python codebase. For this research, the latest 3.9 version of WEKA was downloaded from its official website. After downloading the setup, simple steps were followed to complete the installation process.

Figure 5 shows the creation of the Random Forest machine learning model using the dataset under observation in WEKA.

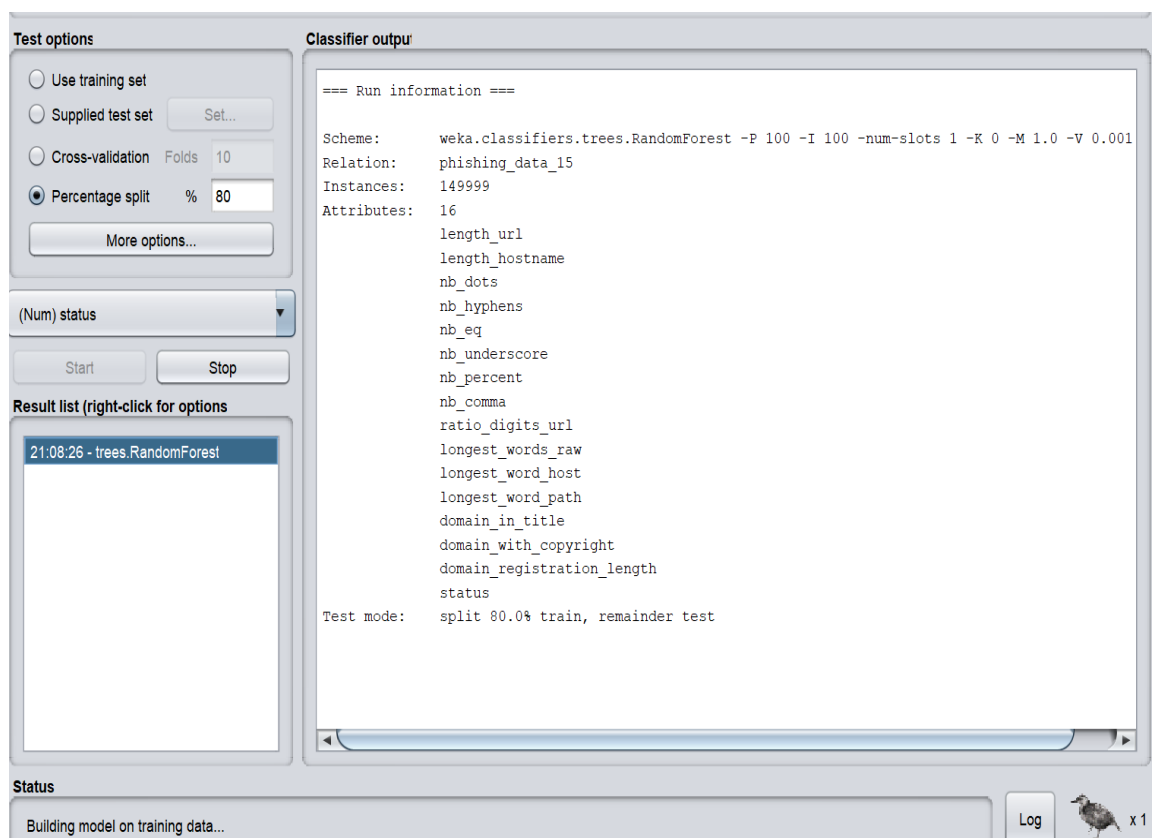


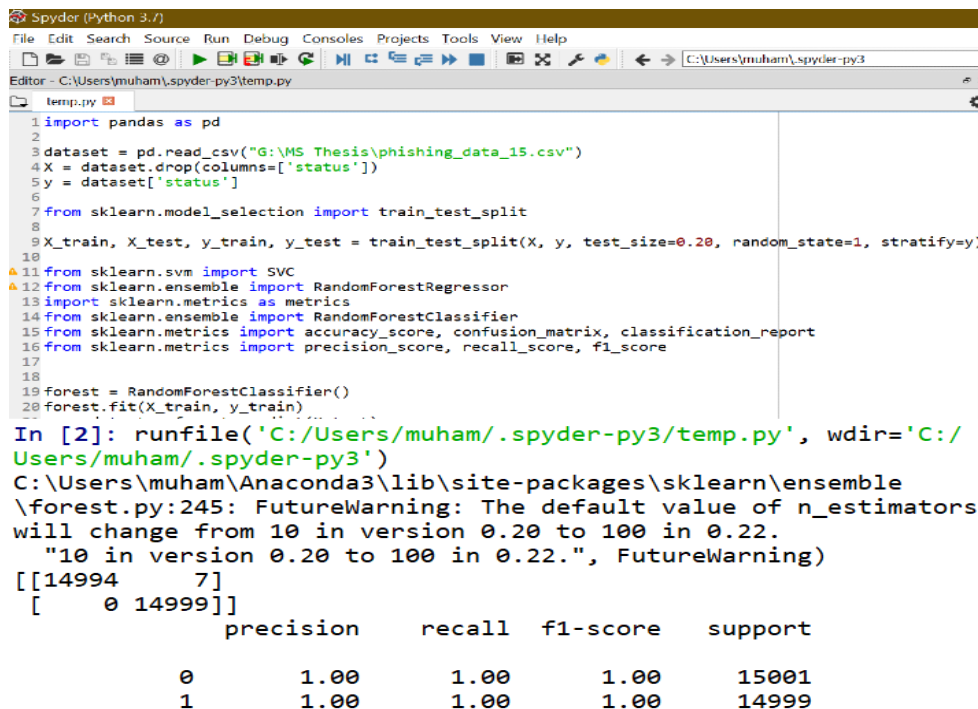
Figure 5: Random Forest Classifier Model in WEKA

4.6 Creating the Python Codebase

Python is a famous programming language that provides a number of useful libraries for implementing machine learning algorithms. The Codebase for this research has been written from scratch using the Python programming language. Since Python is an open-source language, the 3.9 version of Python was downloaded from its official website. After downloading the setup, the required steps for installation were followed, and in the end, the environment variables were added in the windows to enable the Python code to execute smoothly in any IDE.

The built-in libraries, provided by Python, have been used for reading the dataset in the CSV format as well as for creating machine learning models. For importing the CSV file of the relevant dataset, the “pandas” library was used while the “sklearn” built-in library was imported for executing the machine-learning algorithms and evaluating the results.

Figure 6 shows execution of the Random Forest python code in the Spyder IDE to obtain results. The evaluation metrics will be discussed in detail in Chapter 5.



```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor - C:\Users\muham\spyder-py3\temp.py
temp.py
1 import pandas as pd
2
3 dataset = pd.read_csv("G:\MS Thesis\phishing_data_15.csv")
4 X = dataset.drop(Columns=['status'])
5 y = dataset['status']
6
7 from sklearn.model_selection import train_test_split
8
9 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=1, stratify=y)
10
11 from sklearn.svm import SVC
12 from sklearn.ensemble import RandomForestRegressor
13 import sklearn.metrics as metrics
14 from sklearn.ensemble import RandomForestClassifier
15 from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
16 from sklearn.metrics import precision_score, recall_score, f1_score
17
18
19 forest = RandomForestClassifier()
20 forest.fit(X_train, y_train)
In [2]: runfile('C:/Users/muham/.spyder-py3/temp.py', wdir='C:/Users/muham/.spyder-py3')
C:\Users\muham\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
[[14994      7]
 [      0 14999]]
      precision    recall  f1-score   support

0         1.00      1.00      1.00     15001
1         1.00      1.00      1.00     14999

```

Figure 6: Implementation of Random Forest Python Code in Spyder IDE

4.7 Installing Pre-requisite Softwares

Before executing the Python code for this project, there are certain software requirements that need to be fulfilled. These software tools play a crucial role in the smooth execution and functionality of the code. Below is a list of the essential software along with their download links:

- 1) *Archiving tool such as WinRAR*: This software is necessary for extracting and compressing files and folders. WinRAR is a widely used archiving tool that provides efficient file compression and extraction capabilities. WinRAR can be downloaded from their official website at:

<https://www.win-rar.com/start.html?&L=0>

- 2) *Python Interpreter*: It is essential to have a Python interpreter installed on the system to execute the Python code. It is advised to use the latest version of Python for better compatibility and performance. Python's latest version can be downloaded from the official Python website at:

<https://www.python.org/downloads/windows/>

- 3) *Anaconda, Spyder, or any other Python IDE*: An integrated development environment (IDE) is required to perform various operations on the database and execute the Python code seamlessly. Anaconda and Spyder are popular Python IDEs that provide a comprehensive set of tools and features for data analysis and scientific computing.

Anaconda can be downloaded from: <https://www.anaconda.com>

4.8 Summary

This chapter covered the experimental setup and the prerequisites for carrying out the experiments for the study. The collection of the dataset and the installation of the pre-requisite softwares and tools have also been discussed. The results obtained after the experimentations will be discussed in Chapter 5.

Chapter 5

Experimental Results

Chapter 5 sheds light on the results obtained after performing the experiments. A thorough comparison between these results and the benchmark approach will also be provided in this chapter.

5.1 Overview

This research provides a light-weight machine-learning-based anti-phishing solution. The following sections will highlight the effectiveness of the approach by explaining the results obtained after implementing the machine learning algorithms.

5.2 Identification of Optimal Features Set

The selection of appropriate lexical features is a critical aspect of any research in the field of natural language processing. In this study, identifying an optimal set of lexical features was of utmost importance. The reason being, if the set of features is too large, the presence of redundant features could add noise to the final results. Conversely, if the feature set is too restricted, the detection accuracy would not be optimal. Therefore, to overcome these issues, a set of the most relevant lexical features was identified and ranked in order of their significance.

As mentioned in Chapter 4, the initial dataset obtained from Kaggle consisted of 70 extracted features. But it quickly became evident that the feature set contained redundant and less significant features that were only adding to the complexity of the final ML model. By having a glance at the dataset, we observed more than one feature of a similar category, meaning that this feature set was not optimal.

Therefore, in order to obtain the best detection accuracies, it was paramount to identify the most significant features that would yield the best possible outcome when the machine learning model is created using an extended dataset.

In order to identify the most significant features, we employed the Random Forest algorithm in Python to rank all the features in descending order of their significance. In this way, the significance value of each feature was calculated.

Once the significance value against each feature was identified, we removed those features with values less than 0.01. This cut-off value is chosen so that all the lesser contributing feature would be eliminated from the final feature set.

Following various iterations of Python code, the final set of 15 lexical features was identified, and the rest of the features were removed since their contribution to the final results was negligible. For further analysis and evaluation of the proposed methodology, these 15 features have been used.

Figure 7 displays the finalized list of the 15 optimal lexical features that were identified, along with their importance values. Each of these features had a significant impact on the final detection results. Table 4 presents the selected features in a tabular form, making it easier to comprehend and analyze their individual contributions to the final detection results.

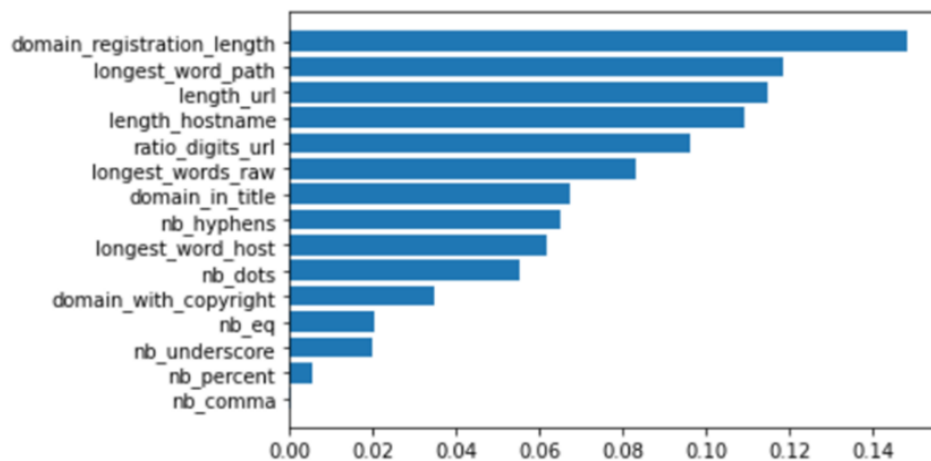


Figure 7: Feature Importance of the Final Selected Set of Features

Table 4: Finalized Set of Features

S No	Feature Name	Symbol/ Explanation	Min Value	Max Value
1	Length of URLs	Overall length of the website URL	13	342
2	Length of hostname	Length of the hostname subsection	4	214
3	Number of dots	.	1	24
4	Number of hyphens	-	0	32
5	Number of “equal to” symbols	=	0	19
6	Number of underscore symbols	–	0	18
7	Number of percent signs	%	0	38
8	Number of comma characters	,	0	4
9	Ratio of number of digits in the URL	No of digits in URL/ length of URL	0	0.622
10	Length of longest word in the URL	Length of the longest word in URL	2	200
11	Length of longest word in the host	Length of the longest word in the host part of the URL	1	62
12	Length of longest word in the path	Length of the longest word in the path section	0	87
13	Domain name in the title of URL	Whether the domain name exists in the URL (1 or 0)	0	1
14	Domain name with copyright	Whether the domain name is registered (1 or 0)	0	1
15	Registration length of domain name	Age of the domain name	0	29829

Before analyzing the final results, it's important to discuss the aforementioned lexical features in detail. As the name suggests, "Length of URLs" indicates the overall length of the website's (either phishing or legitimate) URL. Minimum length of any URL is 13 while the maximum value is 342. "Length of hostname" indicates the length of the hostname section of a URL. If the hostname is unusually long, it generally indicates a phishing URL. Meanwhile, number of dots are also an important factor in detection of phishing URLs. It has been commonly observed that a website with large numbers of dots belongs to the category of phishing websites. Similarly, number of symbols like hyphens "-", equal to "=", underscore "_", percent "%", and comma "," in a website URL can also be used to classify phishing websites. "Ratio of number of digits in the URL" is calculated by dividing the total number of digits in the URL by the overall length of that URL.

Lexical features like "length of the longest word in the URL", "length of the longest word in the host", and "length of the longest word in the path" are self-explanatory. "Domain in the title of URL" has a value of 0 or 1, depending on whether or not a domain name is present in the URL title. Similarly, "Domain name with copyright" has a value of 0 or 1, depending on whether or not the domain name of the URL is officially registered. "Registration length of the domain name" indicates the age of the domain name, i.e., the number of days passed since the domain name of the URL was registered (0 means the domain name is unregistered).

5.3 Evaluating Proposed Method

A number of machine learning models have been employed in order to test the effectiveness of the proposed approach. Nine classification algorithms have been implemented to get a true picture and then the obtained results are compared with the benchmark study. The dataset under observation has been randomly divided into the training and testing sets in the ratio of 80%-20%. It means that for each machine learning algorithm, the model will be trained using 80% entries of the dataset while the testing will be performed on the rest of the 20% of entries.

The results obtained after the implementation of all nine ML algorithms have been highlighted and discussed in the following passage:

Table 5: Proposed Scheme Results on All Nine Classifiers

ML Algorithm	Precision	Recall	F1 Score	Accuracy
Random Forest	99.97%	99.97%	99.97%	99.97%
kNN Classifier	99.98%	99.98%	99.98%	99.98%
Logistic Regression	92.897%	92.896%	92.896%	92.89%
Naïve Bayes (Gaussian)	75.67%	68.43%	65.76%	68.06%
Decision Tree	72.46%	72.465%	72.459%	72.46%
SVM	70%	64%	62%	64.37%
Gradient Booster	85.77%	85.78%	85.77%	85.78%
Adaptive Boosting	84.15%	84.156%	84.139%	84.14%
Extreme Gradient Boosting	85.59%	85.60%	85.59%	85.59%

From table 4, it is evident that the kNN algorithm yields the best accuracy out of all the nine algorithms implemented. On the enlarged dataset, SVM fares the worst in terms of accuracy with only 64.37% accuracy in the classification of benign and phishing websites. kNN classifier also performs the best in terms of true positives and false negatives, producing the best percentages of precision, recall, and F1 score.

There are multiple reasons why kNN has performed better than the rest of the algorithms in this particular scenario. The kNN algorithm is non-parametric, making it suitable for datasets without assumptions about the underlying distribution. It excels in scenarios with unique and independent data entries, providing flexibility in handling diverse datasets. Being a lazy learner, the kNN algorithm avoids constructing a model during training. Instead, it stores the training

CHAPTER 5. EXPERIMENTAL RESULTS

instances and uses them directly during the classification or regression process, making it computationally efficient and allowing for real-time prediction. This makes kNN a good choice for problems where the training data is significantly large. That's why when nine different ML algorithms were used to evaluate the proposed methodology, kNN performed the best with a detection accuracy of 99.98%.

At the other end of the spectrum, SVM is a parametric algorithm, which means that it makes assumptions about the underlying distribution of the data. This can make it less robust to problems where the dataset is large. SVMs are also a greedy learner, which means that they build a model of the data during training. This can make SVM not only more computationally expensive for large datasets, but also less accurate and precise. Therefore, the Support Vector Machine algorithm yielded a detection accuracy of only 64.37%.

Logistic Regression and Random Forest also performed well since the former is able to learn from large datasets and make accurate predictions, while the latter is designed specifically for binary classification tasks, and in this case, the target variable was binary.

Extreme gradient boosting, Gradient boosting, and Adaptive boosting, are powerful machine learning algorithms that are particularly well-suited for problems where the data is complex or noisy. These three algorithms produced almost identical accuracies of approximately 85%, indicating that these boosting algorithms adapted well to the significantly large dataset.

Decision trees work by recursively dividing the input space into regions based on the values of the input features until a decision can be made. The detection accuracy of 72% can be explained by the inappropriate hyperparameters, such as maximum depth or minimum samples per leaf, which play a major role in the overall performance of this algorithm.

Gaussian Naïve Bayes performs the best when the features are not co-related in an ideal scenario. However, the lexical features implored in this study were somewhat linked to each other, resulting in Naïve Bayes yielding an accuracy of only 68.06%.

5.4 Summary

This chapter discussed and analyzed the results achieved during this research. Various metrics evaluated after the implementation of nine ML algorithms were explained in detail. The next chapter will provide validation as well as the comparison of these obtained results with the benchmark approach.

Chapter 6

Discussion and Analysis

This chapter will provide an in-depth analysis and further discussions of the results deduced after the experiments. The previous chapter explained that machine learning models were created using nine classification algorithms and the subsequent results that were collected. In this chapter, the various metrics evaluated in this research will be compared with the benchmark approach.

6.1 Introduction

This section will explain the effectiveness of the proposed research. Nine classification algorithms have been selected and their results are compared with the benchmark approach to evaluate the methodology proposed. In this research, only the most significant lexical features have been employed.

To compare the results of the proposed methodology with the technique used in the reference study, a comprehensive evaluation of the benchmark approach was conducted using an extended dataset. The aim was to assess the performance of nine different classification algorithms when applied to a larger dataset consisting of 150,000 entries. Interestingly, the initial findings revealed a significant drop in accuracy when the benchmark approach was applied to the extended dataset. This unexpected outcome raised concerns about the scalability and effectiveness of the existing technique in handling larger data volumes. It became apparent that relying solely on the original nine lexical features, as chosen by the reference paper's author, was insufficient to achieve accurate predictions on this larger dataset.

CHAPTER 6. DISCUSSION AND ANALYSIS

To overcome this constraint, the proposed methodology aims to augment the existing features by incorporating additional lexical features. A meticulous selection process led to the identification of fifteen optimal lexical features that were considered influential in capturing the underlying patterns within the data. These newly integrated features were expected to significantly enhance the overall performance and accuracy of the classification algorithms employed. By expanding the feature set, the proposed methodology aimed to extract more comprehensive insights and improve the ability of the models to make accurate predictions.

Upon implementing the recreated models using the extended dataset and the proposed methodology, the results were highly promising. Table 8 illustrates the comparative accuracies of the three selected algorithms: Random Forest, kNN, and Logistic Regression. These models achieved remarkably similar accuracies to the benchmark approach when trained on the significantly larger dataset of 150,000 entries.

The obtained results provide compelling evidence for the efficacy of the proposed methodology in attaining similar accuracies to the benchmark approach, even in the presence of a significantly larger and more diverse dataset. This outcome not only validates the effectiveness of the proposed methodology but also underlines its robustness and scalability in handling larger data volumes while maintaining consistent and reliable predictive performance. The ability to achieve comparable accuracies on an extended dataset further reinforces the practical applicability and generalizability of the proposed methodology, instilling confidence in its suitability for real-world scenarios with varying data scales and complexities.

6.2 Performance of Reference Method

The evaluation metrics for the reference methodology are displayed in Table 6. It is evident that when machine learning models are created using the dataset containing 20K URLs, Random Forest performs the best with a detection accuracy of 99.57%. kNN yields 99.04% accuracy while SVM and logistic regression are 97.64% and 95.56% accurate in detection of phishing sites.

Table 6: The Performance of Gupta et. al. [4] Approach on KNN, Random Forest, Logistic Regression, and SVM Classifiers

Approach	Precision (%)	Recall(%)	F1 Score(%)	Accuracy
kNN	98.67%	99.45%	99.06%	99.04%
Random Forest	99.7%	99.46%	99.58%	99.57%
Logistic Regression	94.96%	96.3%	95.625%	95.56%
SVM	96.87%	98.50%	97.68%	97.64%

One of the biggest drawbacks of the approach used in the reference paper is the limited number of URLs in the dataset under observation. It was unclear whether the anti-phishing solution proposed by the author would yield similar results on an extended dataset.

The first step of this study was to test the reference methodology on the enhanced dataset. The existing approach failed in this scenario as the reference methodology was not scaled for creating ML models with such a large number of dataset entries. Table 7 shows that when exposed to an extended dataset, the detection accuracy of all four algorithms employed by the authors dropped significantly. Random forest yielded an accuracy of 99.57% when the dataset under observation is the one used in the reference paper (20k URLs). This detection accuracy drops to 78.17% when the length of the dataset is increased to 150k URLs. The detection accuracy of kNN model reduced to 71.17% from 99.04% when the

proposed methodology was tested on an extended dataset. The accuracy of ML model created using logistic regression dropped from 95.56% to 68.27%. while the implementation of the SVM algorithm on the enhanced dataset resulted in a 44.46% decrease in detection accuracy. On average, the detection accuracies decreased by 29.99%.

There are multiple reasons for this decrease in accuracy when ML models are created using a significantly larger dataset. Most importantly, the larger dataset introduces more diverse and complex patterns, making it challenging for the model to accurately generalize. The proposed method was not well-suited to the complexities added by an enhanced dataset and as a result failed to scale up on the introduction of a much larger dataset. This underlines the need for the optimization of the reference method.

Table 7: Performance Evaluation of Gupta et. al. [4] Results on Existing and New Dataset

ML Classifier	Gupta et. al [4] Features: 9, Dataset: 20k				Gupta et. al [4] Features: 9, Dataset: 150k				
	P	R	F1 Score	Acc.	P	R	F1 Score	Acc.	Degradation
Random Forest	99.70	99.46	99.58	99.57	78.17	78.16	78.16	78.17	21.40%
kNN	98.67	99.45	99.06	99.04	71.98	71.71	71.62	71.71	27.33%
Logistic Regression	94.96	96.30	95.62	95.56	68.28	68.26	68.26	68.27	27.29%
Gaussian Naïve Bayes	-	-	-	-	73.98	60.91	54.41	60.40	-
Decision Tree	-	-	-	-	73.69	64.91	61.04	64.49	-
SVM	96.87	98.50	97.68	97.64	61.00	62.00	53.00	53.00	44.64%
Gradient Booster	-	-	-	-	72.62	72.55	72.55	72.59	-
Ada Boost Classifier	-	-	-	-	71.28	71.28	71.26	71.27	-
Extreme Gradient Boosting Classifier	-	-	-	-	72.34	72.26	72.26	72.30	-
Average	97.55	98.42	97.98	97.95	71.48	69.11	66.95	67.96	29.99%

6.3 Performance Comparison Of Proposed And Reference Methods

The performance comparison of Gupta et. al. [4] and the proposed methodology can be seen in Table 7. Visual comparison has also been provided in Figure 8. As discussed earlier, kNN yields the best comparison in terms of phishing website detection. Most notably, the proposed scheme achieves almost identical accuracies in the implementation of Random Forest, kNN, and Logistic Regression.

From the evaluation of these metrics, it is clear that the methodology proposed in this research outperforms the benchmark approach by producing similar results on a dataset that has been increased by almost 650%.

Table 8: Performance Comparison of Proposed and Gupta et al. [4] Techniques on 150k URLs Dataset

ML Classifier	Gupta et. al [4] Features: 9, Dataset: 150k				Proposed Methodology Features: 15, Dataset: 150k				
	P	R	F1 Score	Acc.	P	R	F1 Score	Acc.	Improvement
Random Forest	78.17	78.16	78.16	78.17	99.97	99.97	99.97	99.97	21.80%
kNN	71.98	71.71	71.62	71.71	99.98	99.98	99.98	99.98	28.27%
Logistic Regression	68.28	68.26	68.26	68.27	92.89	92.89	92.89	92.89	24.62%
Gaussian Naïve Bayes	73.98	60.91	54.41	60.40	75.67	68.43	65.76	68.06	07.66%
Decision Tree	73.69	64.91	61.04	64.49	72.46	72.46	72.45	72.46	07.97%
SVM	61.00	62.00	53.00	53.00	70.00	64.00	62.00	64.37	11.37%
Gradient Booster	72.62	72.55	72.55	72.59	85.77	85.78	85.77	85.78	13.19%
Ada Boost Classifier	71.28	71.28	71.269	71.27	84.15	84.15	84.13	84.14	12.87%
Extreme Gradient Boosting Classifier	72.34	72.26	72.26	72.30	85.59	85.60	85.59	85.59	13.29%
Average	71.48	69.11	66.95	67.96	85.16	83.69	83.17	83.69	15.73%

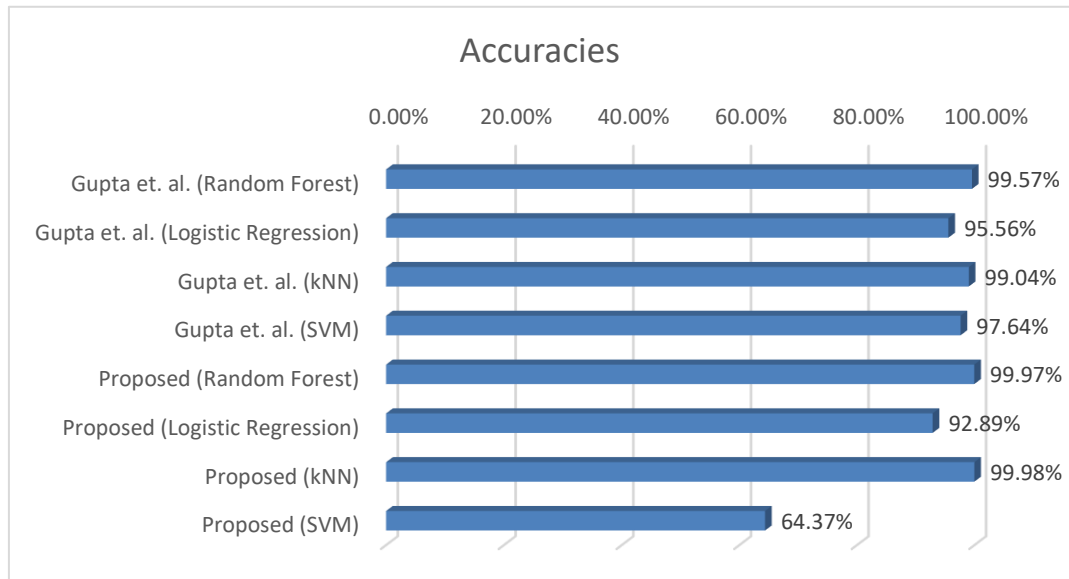


Figure 8: Comparison of Detection Accuracy of the Proposed and Benchmark Approach

Table 8 highlights the comparison of the number of dataset entries and the number of features used in the benchmark approach and this research. While the research by Gupta et. al [4] focused on 9 lexical features and a dataset that contained 20,000 entries, this study goes one step ahead by testing the methodology on a dataset of 150k entries. The metrics were evaluated first by testing the proposed methodology on the 9 features selected in the benchmark approach and then on the optimized set of 15 features.

The introduction of six additional lexical features is one of the key differences between the two approaches. While the detection accuracy of ML models dropped significantly when the length of the dataset under observation was increased, Table 8 shows that the selected combination of 15 features and 1,50,000 dataset entries improves the accuracy of each ML algorithm. This improvement is directly linked to the additional features added to the proposed methodology.

In Table 9, the evaluation metrics obtained after the execution of all nine classifiers have been highlighted. As shown in the table, the kNN classifier performs the best in terms of detection accuracy. Random Forest is not far behind with 99.97% accuracy, while Support Vector Machine (SVM) fares the worst with an

accuracy of less than 65%. The precision, recall, and F1 score values against each of the remaining machine learning algorithms have been provided in the following table.

Table 9: Detection Results Using Proposed Methodology on Various Classification Machine Learning Algorithms

Approach	Precision (%)	Recall (%)	F1 - Score (%)	Accuracy (%)
Random Forest	99.97%	99.97%	99.97%	99.97%
kNN	99.98%	99.98%	99.98%	99.98%
Logistic Regression	92.897%	92.896%	92.896%	92.89%
Gaussian Naïve Bayes	75.67%	68.43%	65.76%	68.06%
Decision Tree	72.46%	72.465%	72.459%	72.46%
SVM	70%	64%	62%	64.37%
Gradient Booster	85.77%	85.78%	85.77%	85.78%
Ada Boost Classifier	84.15%	84.156%	84.139%	84.14%
Extreme Gradient Boosting Classifier	85.59%	85.60%	85.59%	85.59%

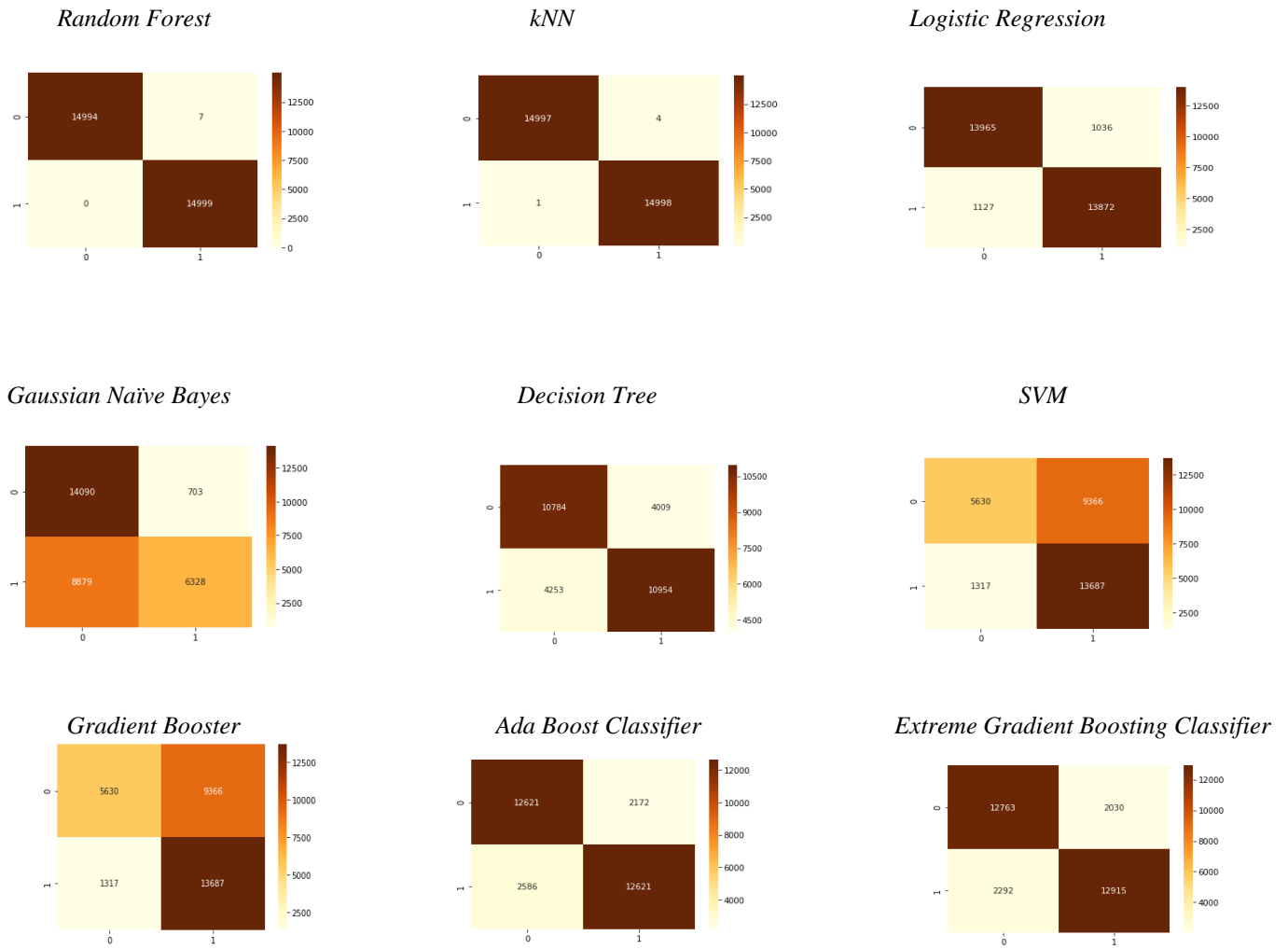


Figure 9: Confusion Matrices of Employed ML Classifiers

“ROC Curve” stands for Receiver Operating Characteristic Curve. It can be defined as a graphical plot that illustrates the relationship between sensitivity and specificity. The ROC graph provides a visual representation of the performance of a test, with the area under the curve (AUC) serving as a measure of its utility. When the curve is closer to the upper left corner, where sensitivity and specificity values are both 1, it indicates higher accuracy. In other words, a larger AUC suggests better test performance in distinguishing between true positives and true negatives.

Figure 10 shows the combined ROC Curves of all nine classifiers. The Random Forest algorithm performs the best in this metric as its AUC value equals one, and

the curve is perpendicular to the X-Axis. The curves of the rest of the ML algorithms can be analyzed in Figure 10.

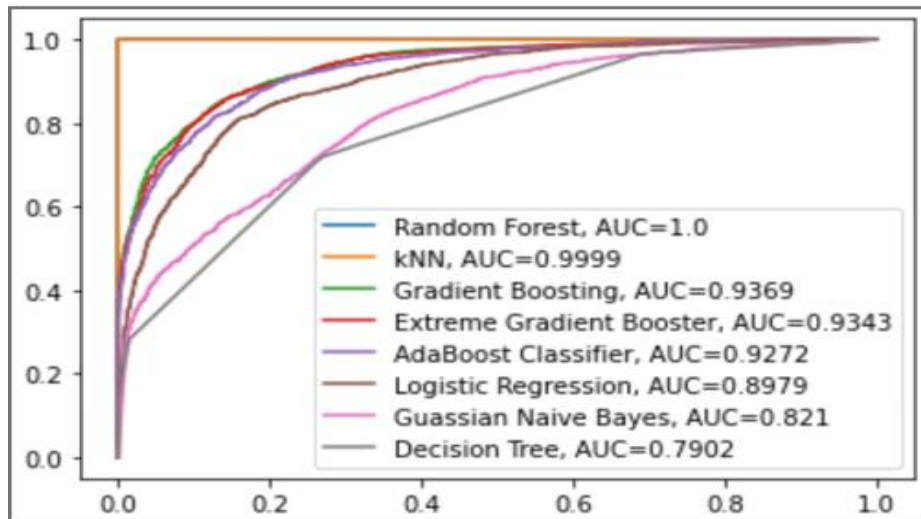


Figure 10: ROC Curves of All Nine Algorithms Implemented

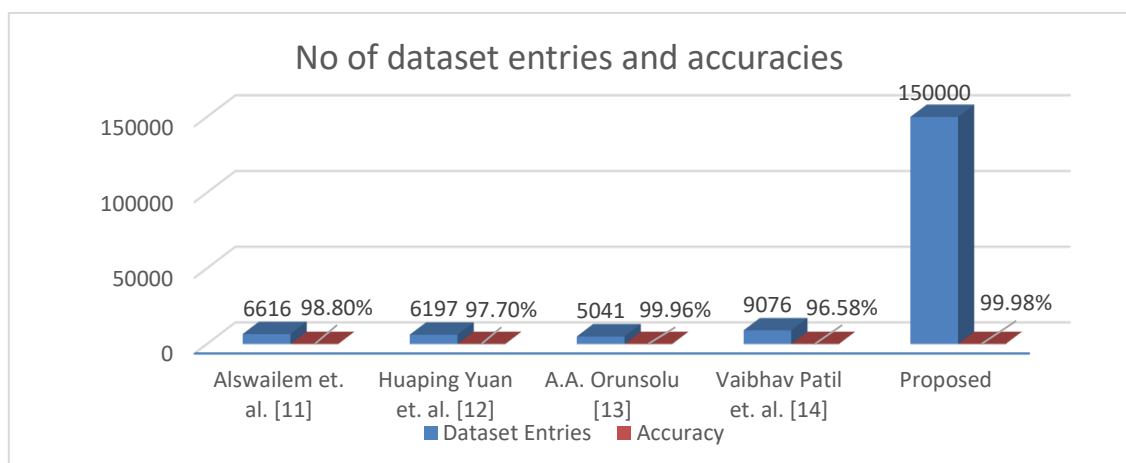


Figure 11: Comparison of No. of Dataset Entries and the Detection Accuracy of Proposed and Existing Approaches

Figure 11 compares the detection accuracies and the length of the dataset of the existing and proposed anti-phishing methodologies in graphical form. Table 10 provides a comprehensive comparison between the proposed methodology and the existing solutions. Previous research done in a similar domain and their detection accuracies have been carefully compiled to give a clear picture. From table 10, it

is evident that even on a significantly larger dataset, the detection accuracies obtained in this study are at par, or in some cases, better than the existing solutions.

Table 10: Performance Comparison of Proposed and Existing Techniques

Relevant Papers	No. of URLs	Employed URLs attributes /features	Random Forest	kNN	Logistic Regression	Naïve Bayes	Decision Tree	SVM	Extreme GB Classifier
Alswailem et. al. [27]	6116	26	98.00 %	-	-	-	-	-	-
Huaping Yuan et. al. [28]	6197	12	96.00 %	92.50 %	95.54%	-	96.30%	-	97.10%
A.A. Orunsolu [29]	5041	15	-	-	-	99.96%	-	99.96 %	-
Vaibhav Patil et. al. [30]	9076	12	96.58 %	-	96.23%	-	96.23%	-	-
Gururaj et. al. [31]	11056	30	96.87 %	93.53 %	-	-	96.05%	48.56 %	
Alyssa Anne et. al. [32]	5126	30		96.20 %	93.00%	-	97.30%	92.00 %	-
Suleiman Y. et. al. [33]	11055	30	97.30 %	-	-	90.70%	-	92.70 %	-
Weiheng Bai et. al. [34]	7058	12	-	-	95.12%	70.05%	90.00%	95.15 %	-

CHAPTER 6. DISCUSSION AND ANALYSIS

Selvakumari et. al. [35]	95911	12	94.40 %	93.10 %	79.00%	-	95.50%	-	93.40%
Ankit Kumar et. al. [37]	2544	13	97.37 %	-	98.42%	95.79%	-	91.47 %	-
Junaid Rashid et. al. [38]	5000	5	-	-	-	-	-	95.66 %	-
Neda Abdelhamid et. al. [39]	11000	5	-	-	-	93%	-	-	-
Mohammad Almseidin et. al. [40]	10000	20	98.11 %	-	-	-	-	-	-
Che-Yu Wu et. al. [41]	15000	14	-	-	58.6%	-	82.3%	92.6%	-
AaishaMakkar et. al. [42]	1740	32	-	96.3%	-	-	-	-	-
Shweta Singh et. al. [43]	73575	10	-	98%	-	-	-	-	-
Abdelhakim et. al. [44]	11430	7	96.83 %	-	-	-	94.13%	73.68 %	-
Uğur et. al. [45]	13791	58	97.91 %	94.36 %	-	83.46%	96.34%	92.77 %	97.88%
A.A.Orunsolu et. al. [46]	5041	15	-	-	-	99.96%	-	99.96 %	-
Noor Zaini et. al. [47]	15000	15	94.79 %	93.08 %	-	-	-	-	-
G. Bottazzi et. al. [48]	86000	53	-	-	-	78%	89.2%	-	-

CHAPTER 6. DISCUSSION AND ANALYSIS

Ammar Odeh et. al. [49]	20000	30	-	-	-	-	-	-	98.9%
Smita Sindhu et. al. [50]	11055	9	97.36 %	-	-	-	-	97.45 %	-
Proposed Methodology	150000	15	99.97 %	99.98 %	92.89%	68.06%	72.46%	64.37 %	85.59%

6.4 Applicability of the Approach

The following cases are penned down to understand the applications and uses of the proposed solution. The proposed approach can help in the early detection of phishing attacks and minimize or prevent financial as well as reputational damages by accurate classification of phishing websites. These points explain the effectiveness of this approach based on the results obtained:

- A. The proposed approach can help detect benign and phishing websites based on the lexical features of URLs.
- B. The proposed approach can be integrated with any system, in use by a casual internet user, to prevent phishing attacks.
- C. The proposed approach can be implemented in high-level corporations and organizations to protect them from deadly phishing attacks and improve their overall security measures.

These findings demonstrate the potential impact of the proposed solution in combating phishing attacks. By enabling accurate detection and prevention of such threats, it plays a crucial role in fostering a safer digital environment for individuals and organizations alike. The research emphasizes the importance of proactive measures and highlights the effectiveness of the proposed approach in enhancing security practices and mitigating the risks linked to phishing attacks. These insights also shed light on the practicality and impact of the solution, paving the way for improved cybersecurity strategies and a heightened level of protection against evolving online threats.

6.5 Summary

This chapter provided a summary of all the important results obtained. The results gathered in this study have been compared with the benchmark approach and the other existing solutions. Some applications of this research have also been explained. The next chapter will underline the conclusion and future work.

Chapter 7

Conclusion & Future Work

This chapter concludes the research and sheds light on the limitations and possible future directions. The research has been briefly explained with the key points highlighted along with the improvements that can still be made.

7.1 Conclusion

Phishing attacks have become increasingly common with the exponential increase in the number of internet users around the world. Machine learning algorithms are used to create models that can detect phishing websites with high accuracy. These machine-learning models depend primarily on the length of the dataset and the selection of appropriate features. This research deals with lexical features of websites' URLs. The websites were collected from several open-source databases and then the relevant lexical features were extracted. The aim of this thesis was to optimize the methodology proposed by the benchmark study by extending the dataset under observation, without compromising the accuracy. Nine different classification algorithms were implemented, with the fifteen most significant lexical features incorporated. The obtained results were compared with the benchmark approach. The comparison revealed that three ML algorithms, Random Forest, kNN, and Logistic Regression, produce accuracies almost identical to the reference study while evaluating a significantly larger dataset. The final results show that kNN performs the best with 99.98% detection accuracy in the classification of phishing websites, better than the benchmark technique.

7.2 Limitation & Future Work

In this study, nine classification algorithms have been employed. However, other existing ML clustering algorithms can be explored for comparison. We also intend to test this methodology on a different dataset with different sets of lexical features. Future studies can also perform dataset validation. Based on the proposed methodology, a browser tool (extension) can also be developed to instantly notify the user if he/she is accessing a phishing site.

Bibliography

- [1] Ogi Djuraskovic, How Many Websites Are There? – The Growth of The Web (1990–2022) [online] Available at: < <https://firstsiteguide.com/how-many-websites/> [Accessed 29 Aug 2022].
- [2] PhishProtection.com. 2022. History of Phishing. [online] Available at: <<https://www.phishprotection.com/resources/history-of-phishing/>.> [Accessed 30 April 2022]
- [3] Maddie Rosenthal, Must-Know Phishing Statistics: Updated 2022. [online] Available at: < <https://www.tessian.com/blog/phishing-statistics-2020/>> [Accessed 26 April 2022].
- [4] Gupta, B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A., & Chang, X. (2021). A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. *Computer Communications*, 175, 47-57. <https://doi.org/10.1016/j.comcom.2021.04.023>
- [5] Arachchilage, N. and Love, S., 2014. Security awareness of computer users: A phishing threat avoidance perspective. *Computers in Human Behavior*, 38, pp.304-312.
- [6] S. Afroz, R. Greenstadt, Phishzoo: Detecting phishing websites by looking at them, in: *Fifth IEEE International Conference on Semantic Computing*, 201
- [7] Dong, X., Clark, J. A., & Jacob, J. (2008, May 1). Modelling user-phishing interaction. *IEEE Xplore*. <https://doi.org/10.1109/HSI.2008.4581513>
- [8] Kirlappos, I., & Sasse, M. A. (2012). Security Education against Phishing: A Modest Proposal for a Major Rethink. *IEEE Security & Privacy Magazine*, 10(2), 24–32. <https://doi.org/10.1109/msp.2011.179>
- [9] Alsharnouby, M., Alaca, F., & Chiasson, S. (2015). Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies*, 82, 69–82.

BIBLIOGRAPHY

<https://doi.org/10.1016/j.ijhcs.2015.05.005>

- [10] Patil, S. and Dhage, S. 2019. A Methodical Overview on Phishing Detection along with an Organized Way to Construct an Anti-Phishing Framework. 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)
- [11] Kunju, M., Dainel, E., Anthony, H. and Bhelwa, S., 2019. Evaluation of Phishing Techniques Based on Machine Learning. 2019 International Conference on Intelligent Computing and Control Systems (ICCS)
- [12] Rao, R. S., & Pais, A. R. (2017). An Enhanced Blacklist Method to Detect Phishing Websites. *Information Systems Security*, 323–333. https://doi.org/10.1007/978-3-319-72598-7_20
- [13] Bell, S., & Komisarczuk, P. (2020). An Analysis of Phishing Blacklists: Google Safe Browsing, OpenPhish, and PhishTank. *Proceedings of the Australasian Computer Science Week Multiconference*. <https://doi.org/10.1145/3373017.3373020>
- [14] Sheng, S., Wardman, B., Warner, G., Cranor, L., Hong, J., & Zhang, C. (2009). An empirical analysis of phishing blacklists.
- [15] Jain, A., & Gupta, B. (2017). Phishing Detection: Analysis of Visual Similarity Based Approaches. *Security And Communication Networks*, 2017, 1-20. <https://doi.org/10.1155/2017/5421046>
- [16] Al-Ahmadi, S., & Alharbi, Y. (2020). A Deep Learning Technique for Web Phishing Detection Combined URL Features and Visual Similarity. *International Journal Of Computer Networks & Communications*, 12(5), 41-54. <https://doi.org/10.5121/ijcnc.2020.12503>
- [17] Zhou, Y., Zhang, Y., Xiao, J., Wang, Y., & Lin, W. (2014). Visual Similarity Based Anti-phishing with the Combination of Local and Global Features. *Trust, Security and Privacy in Computing and Communications*. <https://doi.org/10.1109/trustcom.2014.28>
- [18] Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012). How Many Trees in a Random Forest? *Machine Learning and Data Mining in Pattern Recognition*, 7376, 154–168. https://doi.org/10.1007/978-3-642-31537-4_13

BIBLIOGRAPHY

- [19] Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia Medica*, 24(1), 12–18. <https://doi.org/10.11613/bm.2014.003>
- [20] Beckmann, M., Ebecken, N. F. F., & Pires de Lima, B. S. L. (2015). A KNN Undersampling Approach for Data Balancing. *Journal of Intelligent Learning Systems and Applications*, 07(04), 104–116. <https://doi.org/10.4236/jilsa.2015.74010>
- [21] Winters-Hilt, S., & Merat, S. (2007). SVM clustering. *BMC Bioinformatics*, 8(S7). <https://doi.org/10.1186/1471-2105-8-s7-s18>
- [22] ZHANG, H. (2005). EXPLORING CONDITIONS FOR THE OPTIMALITY OF NAÏVE BAYES. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(02), 183–198. <https://doi.org/10.1142/s0218001405003983>
- [23] Somvanshi, M., Chavan, P., Tambade, S., & Shinde, S. V. (2016, August 1). A review of machine learning techniques using decision tree and support vector machine. *IEEE Xplore*. <https://doi.org/10.1109/ICCUBEA.2016.7860040>
- [24] Schapire, R. E. (2013). Explaining AdaBoost. *Empirical Inference*, 37–52. https://doi.org/10.1007/978-3-642-41136-6_5
- [25] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems*, 30. <https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bd9eb6b76fa-Abstract.html>
- [26] Bansal, A., & Kaur, S. (2018). Extreme Gradient Boosting Based Tuning for Classification in Intrusion Detection Systems. *Communications in Computer and Information Science*, 372–380. https://doi.org/10.1007/978-981-13-1810-8_37
- [27] Alswailem, A., Alabdullah, B., Alrumayh, N. and Alsedrani, A., 2019. Detecting Phishing Websites Using Machine Learning. 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS).
- [28] Yuan, H., Chen, X., Li, Y., Yang, Z. and Liu, W., 2018. Detecting

BIBLIOGRAPHY

- Phishing Websites and Targets Based on URLs and Webpage Links. 2018 24th International Conference on Pattern Recognition (ICPR).
- [29] Orunsolu, A., Sodiya, A. and Akinwale, A., 2022. A predictive model for phishing detection. *Journal of King Saud University - Computer and Information Sciences*, 34(2), pp.232-247.
- [30] Patil, V., Thakkar, P., Shah, C., Bhat, T., & Godse, S. (2018). Detection and Prevention of Phishing Websites Using Machine Learning Approach. 2018 Fourth International Conference On Computing Communication Control And Automation (ICCUBEA). <https://doi.org/10.1109/iccubea.2018.86974>
- [31] Harinahalli Lokesh, G., & BoreGowda, G. (2020). Phishing website detection based on effective machine learning approach. *Journal of Cyber Security Technology*, 1–14. <https://doi.org/10.1080/23742917.2020.1813396>
- [32] Ubung, A. A., Kamilia, S., Abdullah, A., Jhanjhi, N., & Supramaniam, M. (2019). Phishing Website Detection: An Improved Accuracy through Feature Selection and Ensemble Learning. *International Journal of Advanced Computer Science and Applications*, 10(1). <https://doi.org/10.14569/ijacsa.2019.0100133>
- [33] Yerima, S. Y., & Alzaylaee, M. K. (2020). High Accuracy Phishing Detection Based on Convolutional Neural Networks. 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS). <https://doi.org/10.1109/iccais48893.2020.9096869>
- [34] Bai, W. (2020, August 1). Phishing Website Detection Based on Machine Learning Algorithm. IEEE Xplore. <https://doi.org/10.1109/CDS49703.2020.00064>
- [35] Selvakumari, M., Sowjanya, M., Das, S., & Padmavathi, S. (2021). Phishing website detection using machine learning and deep learning techniques. *Journal of Physics: Conference Series*, 1916(1), 012169. <https://doi.org/10.1088/1742-6596/1916/1/012169>
- [36] Proposed collected Phishing URLs Dataset https://drive.google.com/file/d/1WyF8aI_h_1A5fdiiv7R5ATEq47IawJn2/

BIBLIOGRAPHY

[view?usp=sharing](#)

- [37] Jain, A. K., & Gupta, B. B. (2018). A machine learning based approach for phishing detection using hyperlinks information. *Journal of Ambient Intelligence and Humanized Computing*, 10(5), 2015–2028. <https://doi.org/10.1007/s12652-018-0798-z>
- [38] Rashid, J., Mahmood, T., Nisar, M. W., & Nazir, T. (2020). Phishing Detection Using Machine Learning Technique. *IEEE Xplore*. <https://doi.org/10.1109/SMART-TECH49988.2020.00026>
- [39] Abdelhamid, N., Thabtah, F., & Abdel-jaber, H. (2017). Phishing detection: A recent intelligent machine learning comparison based on models content and features. 2017 IEEE International Conference on Intelligence and Security Informatics (ISI). <https://doi.org/10.1109/isi.2017.8004877>
- [40] Almseidin, M., Zuraiq, A. A., Al-kasassbeh, M., & Alnidami, N. (2019). Phishing Detection Based on Machine Learning and Feature Selection Methods. In www.learntechlib.org. International Association of Online Engineering. <https://www.learntechlib.org/p/216410/>
- [41] Wu, C.-Y., Kuo, C.-C., & Yang, C.-S. (2019, August 1). A Phishing Detection System based on Machine Learning. *IEEE Xplore*. <https://doi.org/10.1109/ICEA.2019.8858325>
- [42] Makkar, A., & Kumar, N. (2021). PROTECTOR: An optimized deep learning-based framework for image spam detection and prevention. *Future Generation Computer Systems*, 125, 41–58. <https://doi.org/10.1016/j.future.2021.06.026>
- [43] Singh, S., Singh, M. P., & Pandey, R. (2020, October 1). Phishing Detection from URLs Using Deep Learning Approach. *IEEE Xplore*. <https://doi.org/10.1109/ICCCS49678.2020.9277459>
- [44] Hannousse, A., & Yahiouche, S. (2021). Towards benchmark datasets for machine learning based website phishing detection: An experimental study. *Engineering Applications of Artificial Intelligence*, 104, 104347. <https://doi.org/10.1016/j.engappai.2021.104347>

BIBLIOGRAPHY

- [45] Ozker, U., & Sahingoz, O. K. (2020). Content Based Phishing Detection with Machine Learning. 2020 International Conference on Electrical Engineering (ICEE). <https://doi.org/10.1109/icee49691.2020.9249892>
- [46] Orunsolu, A. A., Sodiya, A. S., & Akinwale, A. T. (2019). A predictive model for phishing detection. Journal of King Saud University - Computer and Information Sciences. <https://doi.org/10.1016/j.jksuci.2019.12.005>
- [47] Zaini, N. S., Stiawan, D., Razak, M. F. A., Firdaus, A., Wan Din, W. I. S., Kasim, S., & Sutikno, T. (2020). Phishing detection system using machine learning classifiers. Indonesian Journal of Electrical Engineering and Computer Science, 17(3), 1165. <https://doi.org/10.11591/ijeecs.v17.i3.pp1165-1171>
- [48] Bottazzi, G., Casalicchio, E., Cingolani, D., Marturana, F., & Piu, M. (2015). MP-Shield: A Framework for Phishing Detection in Mobile Devices. 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing. <https://doi.org/10.1109/cit/iucc/dasc/picom.2015.293>
- [49] Odeh, A., & Keshta, I. (2021). PhiBoost- A novel phishing detection model Using Adaptive Boosting approach. Jordanian Journal of Computers and Information Technology, 7(1), 64. <https://doi.org/10.5455/jcit.71-1600061738>
- [50] Sindhu, S., Patil, S. P., Sreevalsan, A., Rahman, F., & N, M. S. A. (2020). Phishing Detection using Random Forest, SVM and Neural Network with Backpropagation. IEEE Xplore. <https://doi.org/10.1109/ICSTCEE49637.2020.9277256>
- [51] Thilagaraj, T., & Sengottaiyan, N. (2019). Implementation of Classification Algorithms in Educational Data using Weka Tool. International Journal of Computer Sciences and Engineering, 7(5), 1253–1257. <https://doi.org/10.26438/ijcse/v7i5.12531257>
- [52] Stock, E. M., Stamey, J. D., Sankaranarayanan, R., Young, D. M., Muwonge, R., & Arbyn, M. (2012). Estimation of disease prevalence, true positive rate, and false positive rate of two screening tests when disease

BIBLIOGRAPHY

verification is applied on only screen-positives: A hierarchical model using multi-center data. *Cancer Epidemiology*, 36(2), 153–160. <https://doi.org/10.1016/j.canep.2011.07.001>

- [53] Ahuja, R., Chug, A., Gupta, S., Ahuja, P., & Kohli, S. (2019). Classification and Clustering Algorithms of Machine Learning with their Applications. *Nature-Inspired Computation in Data Mining and Machine Learning*, 225–248. https://doi.org/10.1007/978-3-030-28553-1_11