Malicious URL Detection Using Machine Learning and Deep Learning Algorithms

Prepared by

Mahmuda Haque Mumu

Student ID: 2019-1-50-053

&

Tanzina Aishy

Student ID: 2019-1-50-017

Supervised by

Dr. Mohammad Arifuzzaman

Chairperson, Associate Professor

This Thesis Paper is Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Information and Communications Engineering

DEPARTMENT OF ELECTRONICS & COMMUNICATIONS ENGINEERING

EAST WEST UNIVERSITY

APPROVAL

The thesis paper titled "Malicious URL Detection Using Machine Learning and Deep Learning Algorithms" submitted by Mahmuda Haque Mumu (Student ID: 2019-1-50-053) and Tanzina Aishy (Student ID: 2019-1-50-017) to the Department of Electronics and Communications Engineering, East West University, Dhaka, Bangladesh has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Information and Communications Engineering and approved as to its style and contents.

Approved By

(Supervisor)

Dr. Mohammad Arifuzzaman

Chairperson, Associate Professor

ECE Department

East West University

Dhaka, Bangladesh

(Chairperson)

Dr. Mohammad Arifuzzaman

Chairperson, Associate Professor

ECE Department

East West University

Dhaka, Bangladesh

DECLERATION

We declare that our work has not been previously submitted and approved for the award of a degree by this or any other University. As per of my knowledge and belief, this paper contains no material previously published or written by another person except where due reference is made in the paper itself. We hereby, declare that the work presented in this thesis paper is the outcome of the investigation performed by us under the supervision of Dr. Mohammad Arifuzzaman, Associate Professor, Department of Electronics & Communications Engineering, East West University, Dhaka, Bangladesh.

Countersigned

(Supervisor)

Dr. Mohammad Arifuzzaman

Signature

Mahmuda Haque Mumu

Student ID: 2019-1-50-053

Signature

Tanzina Aishy

Student ID: 2019-1-50-017

DEDICATION

This paper is dedicated
To
Our beloved parents and honorable teachers

ACKNOWLEDGEMENT

We would like to convey our sincere gratitude to our supervisor, Dr. Mohammad Arifuzzaman for his valuable advice and sincere guidance to complete this work in an efficient way. His constant support gave us the assurance we needed to work on this paper. We would also like to thank him for his friendship, empathy, and great sense of humor. Through our research, we learned from him many valuable lessons and concepts about machine learning and deep learning techniques. Our completion of this paper could not have been accomplished without the support of our honorable supervisor. We are truly honored and thankful to have him as a supervisor in the journey of our thesis. We are also grateful to our honorable faculty members and office stuffs who rendered their help during the period of our thesis work.

ABSTRACT

The majority of commercial enterprises rely on the World Wide Web's Internet services (WWW). These services are vulnerable to cyberattacks. The majority of cyberattacks occur when users click on malicious URLs. URL is an abbreviation for Uniform Resource Locator, which is the global address of documents and other resources on the World Wide Web. Malicious URLs are compromised URLs that are used for cyberattacks. URLs are used to access legitimate resources on the WWW. When used for other purposes, they endanger data availability, controllability, confidentiality, and integrity. The distribution of malware, illegal information, or illegal images via computers or networks is an example of cybercrime involving computer use to commit other crimes. Malicious URLs can be found in a variety of places, including pop-up windows, social media posts, emails, and texts. These links are created and shared by scammers in an effort to deceive users into clicking. People can be exposed to harmful software, viruses, and other dangerous stuff once they click on these websites. The proposed work in this paper considers multiclass Malicious URL detection and investigates the evaluation metrics of various Machine Learning and Deep Learning classifiers where the experimental results show that the highest performance of the Machine Learning Algorithm and Deep Learning Algorithm is Random Forest Classifier and Convolutional Neural Network (CNN) respectively. The experimental findings also indicate that the suggested URL features and behaviour can considerably increase the capacity to recognize malicious URLs. This implies that the suggested methodology might be regarded as an effective method of identifying malicious URLs.

Table of Contents

List of Figures

List of Tables

CHAPTER ONE

Introduction

1.1 Introduction

In today's world, everything revolves around the internet. The internet is a big part of everything that we do and learn. Without internet, our lives are unimaginable. Every individual, professional, or business operating today is highly dependent on the internet. We have seen that websites are a necessary for businesses. Millions of people, companies, and organizations own domain names to make their online resources accessible to everyone. The World Wide Web significantly facilitated daily life. Using the internet and the web, anyone may now exchange all types of information and content. Many of us won't be able to imagine our lives without the internet because the World Wide Web has become an essential part of our daily living. A lot of information is kept in the form of data on the WWW, a type of virtual space. On the internet, you can post data, photographs, videos, and documents in a planned, organized fashion. Uniform resource locators, or URLs, which are used to identify resources, further identify all of these data. The protocol and domain name are just two of the pieces of a URL that instruct a web browser how and where to get a resource. There are around 1.14 billion websites in existence right now. Around 252,000 new websites are made every day, with 17% of them being active, 83% being inactive. As the Internet advances and expands, more and more of our activities—including e-commerce, business, social networking, and banking—are now carried out online which is increasing the risk of online crime. So, securing the world wide web is becoming increasingly important.

Although protocols and laws protect the connection between the client and server, attackers with malicious intent can still exploit it. The term ‘‘Malicious’’ is a general term for attack types that include phishing, spam and malware, defacement and more.

The process of detecting and blocking potentially dangerous URLs that have the ability to distribute malware, steal personal information, or engage in other criminal actions is known as malicious URL detection. In this world, cyberattack occurs every 39 seconds. Every day, an estimated 30,000 websites are hacked globally, which is projected to cost the US economy $3.5 billion annually. So now it is important to protect the internet. Numerous studies have been conducted recently in an effort to identify methods of malware attack prevention and detection. Among these, machine learning and deep learning algorithms are increasingly

being used to improve the accuracy and efficiency of malicious URL detection systems. These technique allow the detection system to learn from examples of both benign and malicious URLs, and to adapt and improve over time. In this study, we have tried to detect the malicious url by applying some Machine Learning Algorithms and Deep Learning Algorithm **[1].**

We have collected a dataset which consists of huge number of URL's which consists of malicious, benign, phishing and defacement URLs, then we divide the collected dataset into two subsets in the ratio of 80:20 for training purposes and testing purposes, extract features and train the system using the training data and Machine Learning algorithms like Decision Tree algorithm, Random Forest algorithm, KNN Gaussian Naive Bayes algorithm and Deep Learning Algorithm like CNN. At last, we test the system by providing test data and calculating the accuracy using each of the algorithms. Our goal is to present an in-depth analysis on the detection of malicious URLs using machine learning and deep learning algorithms.

1.2 Problem Statement

We use the internet daily to conduct business, check Facebook, and look up fresh information. Malicious urls are frequently seen. Links created to spread scams, attacks, and frauds are known as malicious URLs, commonly referred to as "infected links," "viral links," or simply "weaponized links." URL injections are straightforward but efficient. Hackers insert malicious URLs into web pages, sometimes even seizing control of entire pages, targeting platforms like Wordpress (which runs 60% of today's blogs). When you use your web browser to access a page like this, computer code that installs malware, reroutes you to other dangerous websites, or scrapes personal data from you is executed on your computer. Malicious redirects and browser hijackers, which compel you to visit other malicious websites, are other ways that this is accomplished.

No of the technique, the goal of all these hacking techniques set up on malicious websites is to get you to reveal your data. It might be financial or personal information.Whatever they do, these hackers are attempting to steal from you in order to enrich themselves. Because of this, identifying bad urls is crucial to protecting against these attacks. **[2**]

Researchers are trying to find out the most efficient way to detect malicious url. We are also trying to detect the malicious url using machine learning and deep learning methods. Our experiments follows-

1. Selecting relevant dataset

2. Analysing the dataset using machine learning.

3. Analysing the dataset using deep learning.

4. Presenting comparison between the results after applying these two techniques

5. Analyzing the results and give a conclusion from our findings of this research

1.3 Motivation

Cybersecurity is seriously threatened by rogue websites or URLs. Malicious URLs host unsolicited content (spam, phishing, drive-by downloads, etc.) and deceive users into falling for scams (including financial loss, identity theft, and malware installation), resulting in billions of dollars in damages each year. Some Examples of malicious attacks are-

- The cybercrime organization BAHMUT created fake news websites by stealing the headlines from actual news sources and used them to launch phishing campaigns against consumers, public servants, and companies. **[3]** Links on these malicious websites led users to phishing websites that asked them for their Google, Yahoo, Microsoft, and other login information.
- A major credit bureau suffered a data breach in 2017, exposing approximately 150 million people's personal data. After the bureau's settlement claims website went live two years later, cybercriminals started creating imitation websites in an effort to steal personally identifying information. **[4]**
- By developing bogus websites that looked like authentic coronavirus dashboards, cybercriminals have tried to profit from the COVID-19 outbreak. **[5]** To stay informed about the epidemic, these websites would encourage visitors to download an

app, which would then infect their computers with AZORult malware. Among other things, this malware is used to steal surfing data, cookies, passwords, and bitcoin.

We decide to work in this area because it is crucial to distinguish between harmful and legitimate urls. In order to construct machine learning-based models and deep learning-based models to identify harmful urls so that we can block them in advance before they infect computer systems or propagate across the internet, we have gathered the dataset to include a huge number of samples of malicious URLs.

1.4 Limitations

There are some limitations in our research. This research will not cover all the detection techniques. We use several algorithms in our detection techniques but we will not try all the algorithm for detection techniques. No model will give us the 100% accuracy. However, in our research we will try to focus on the important techniques and features of malicious URL detection.

CHAPTER TWO

Related Works

The main aim of malicious URL detection is to distinguish malicious URLs from benign URLs. In the past few years, research efforts have been made on the detection of malicious URLs using data mining approaches.

Eshete et al. presented a lightweight approach, called BINSPECT **[6]** that combines static analysis and emulation to apply supervised learning techniques in detecting malicious webpages. The experimental evaluation of INSPECT achieved above 97% accuracy with low false signals.

Ma et al. **[7]** explored statistical methods from machine learning classifiers to detect malicious URLs based on lexical and host-based features of URLs. According to their experimental results, classifiers obtained 95-99% accuracy. Although this work achieves high detection accuracy, the extraction of host-based features is time-consuming which can cause a delay in real-time systems.

Curtsinger. et al. proposed ZOZZLE [**8**], a low-overhead solution for detecting and preventing JavaScript malware. ZOZZLE uses a Bayesian classification of a hierarchical feature of the JavaScript abstract syntax tree to predict malware. According to their eexperimentalevaluation on 1.2 million benign JavaScript samples, ZOZZLE achieved a low false-positive rate of 0.0003%.

Abdelhamid et al. [**9**] proposed multi-label classifier-based associative classification (MCAC) to detect phishing websites. MCAC generates single and multi-label rules from a phishing training dataset to classify websites into legitimate or phishy.

Jeeva and Rajsingh [**10**] proposed an intelligent phishing URL detection model based on association rule mining. Fourteen URL features were exposed to associative rule mining apriori algorithm and predictive a priori algorithm. A URL is identified by using association rules in which the features are extracted to acquire unknown knowledge. Strong rules generated by the apriori algorithm with 100% confidence and by the predictive apriori

algorithm with an accuracy level above 99% were considered for further analysis. Features such as transport layer security (TLS), unavailability of the top-level domain in the URL, and keyword within the path portion of the URL were frequent in phishing URLs. The experimental results show that the apriori algorithm mines rule faster than the predictive apriori algorithm. 93% of the phishing URLs are detected using the rules obtained by the a priori algorithm.

Kim et al. [**11**] proposed WebMon detects hidden exploit codes by tracing linked URLs to determine whether the websites in question are malicious. The authors focus on the features of Exploit Kits (EKs) in this paper and introduce 11 feature classes for detecting malicious web pages. They put six supervised learning classification algorithms to the test (random forest, naive Bayes, logistic regression, Bayes net, J48, and SVM). The random forest learning algorithm produced the best results, so WebMon is built with it. According to their experimental results, WebMon has a 98% accuracy rate and is 7.6 times faster than traditional malicious webpage detection tools.

Li et al. [**12**] presented a stacking model for detecting phishing webpages using URL and HTML features that combined gradient boosting decision tree, XGBoost, and LightGBM in multiple layers.They evaluated their model on 50,000 web pages and achieved an accuracy of 97.30%, 4.46% on missing rate, and 1.61% on false alarm rate.

Blacklist approaches such as Google Safe Browsing [**13**] use a list of predefined malicious URLs list to detect malicious URLs. Cao et al. [**14**] proposed an anti-phishing tool based on legitimate login user interfaces (Luis) of websites, called an automated individual white-list (AIWL). AIWL employs the naive Bayes classifier to maintain a white list of trusted websites visited by users and issues alerts whenever an attack is suspected. The drawback of the blacklist-whitelist-based approach is that it cannot proactively detect new malicious web pages.

Liu and Zhang [**15**] proposed a two-wheeled phishing page check method. The first round checks the domain, URL and email of the current page, and if it exceeds the threshold, it is directly identified as a phishing site. Passwords, links and images are checked if they do not pass the second round. If all checks do not exceed the threshold, it is a normal page. However, this method is only used in the financial sector.

Shekokar et al. [**16]** proposed a two-stage phishing page detection scheme. The first step uses the LinkGuard algorithm to analyze the difference between visual links (links displayed by the browser) and real links (hidden in HTML). The second step compares the similarity between snapshots of suspicious sites and legitimate sites by computing the discrete cosine transform. Although this method does not need to maintain a large database of malicious websites, it cannot detect unknown malicious URLs because the rule generation is based on existing malicious URLs.

Researchers have used machine learning techniques for Malicious URL Detection to address these issues over the last decade [**17,18**]. Machine Learning approaches use a set of URLs as training data and learn a prediction function to classify a URL as malicious or benign based on statistical properties. Unlike blacklisting methods, this allows them to generalize to new URLs. The presence of training data is the most important requirement for training a machine learning model. This would correspond to a large number of URLs in the context of malicious URL detection. Machine learning can be broadly divided into three types: supervised, unsupervised, and semi-supervised, which correspond to having labels for the training data, not having labels, and having labels for a subset of the training data, respectively. Labels represent the knowledge of whether a URL is malicious or benign.

CHAPTER THREE

Introduction to Malicious URL

3.1 What is a URL? What is a Malicious URL?

URL is an abbreviation for Uniform Resource Locator, which is the global address of documents and other resources on the World Wide Web. A URL is made up of two main parts:

(i) protocol identifier (indicates what protocol to use)

(ii) resource name (specifies the IP address or the domain name where the resource is located).

A colon and two forward slashes separate the protocol identifier from the resource name. Malicious URLs are compromised URLs that are used for cyberattacks. Indeed, it was discovered that nearly one-third of all websites are potentially malicious in nature **[19]**, demonstrating the widespread use of malicious URLs to commit cybercrime. To launch attacks, a malicious URL or website hosts various unsolicited content in the form of spam, phishing, or a drive-by download.

Not all malicious websites use the same template. While the phrase "malicious site" may conjure up images of a dozen pop-up ads, flashing red alerts, or a window that says "All FiLeS ArE Belong To Us." You're not entirely wrong in your assumptions, but you're overlooking a lot of malicious URL categories if that's all you can think of when it comes to cyber threats.

Webshrinker's ability to identify malicious sites is technically distinct from its ability to identify "normal" sites. Before assigning a category to a site, Webshrinker performs a series of checks to determine whether the site is malicious before assigning it a category such as "gambling." As a result, if a site is found to be malicious, it will only be assigned a "deceptive" value and will not be classified as a non-malicious site.

For an overview of the types of malicious categories, Webshrinker can find the list which is below :

- **Botnet**: These are Command and Control, botnet hosts. Blocking these sites prevent you from receiving commands from already-infected machines.
- **Cryptomining**: Sites that server files or host applications that force your web browser to mine cryptocurrency. This can use several resources.
- **Malware:** Malware is a parent category for a lot of different types of malicious software including ransomware, viruses, spyware, and trojans.
- **Phishing scams**: Websites trick you into handing over personal data. (We have a great blog about phishing on our parent company's site). **[20]**

3.2 Position of Malicious URLs in cybercrime

Cybercriminals who target computers may infect them with malware to damage or disable them. Malware may also be used to delete or steal data. A Denial-of-Service (DoS) attack occurs when cybercriminals prevent users from using a website or network or prevent a business from providing a software service to its customers. [**21**]

The majority of commercial enterprises rely on the World Wide Web's Internet services (WWW). These services, however, are vulnerable to cyberattacks. The majority of cyberattacks occur when users click on malicious URLs. URLs are used to access legitimate resources on the WWW. When used for other purposes, they endanger data availability, controllability, confidentiality, and integrity. The distribution of malware, illegal information, or illegal images via computers or networks is an example of cybercrime involving computer use to commit other crimes.

The WannaCry ransomware attack, a global cybercrime committed in May 2017, was a well-known example of a malware attack. WannaCry is a type of ransomware malware used to extort money by encrypting the victim's data or device. The ransomware exploited a vulnerability in Microsoft Windows computers.

When the WannaCry ransomware attack struck, 230,000 computers in 150 countries were affected. Users were locked out of their files and were sent a message demanding a Bitcoin ransom to regain access. The WannaCry cybercrime is estimated to have cost the world $4 billion in financial losses. The attack is still remembered for its sheer size and scope. [**21**]

3.3 Indicators for Malicious URL

A file on a computer can be found by providing its filename, a URL can be used to track down any website. It is the web address of a WWW resource. Each URL is made up of two main parts. The protocol is the first. HTTPS is the protocol identifier for the URL https://www.google.com. HTTPS is a secure version of the Hypertext Transfer Protocol that is used to retrieve hypertext documents. File Transfer Protocol (FTP), Domain Name System (DNS), and other protocols are also available. The second element is the Resource identifier. The resource name for the URL https://www.google.com is www.google.com. The address of a webpage on the internet is the resource identifier. The proposed work in this paper considers bad URL detection and investigates the evaluation metrics of various Machine Learning classifiers. **[22**]

Scenario: This scenario consists of an indicator for the URL http://x4z9arb.cn/4712/, which is known to be malicious. Unlike the C2 markup and the malware hashing idiom, in this case, the organization creating the indicator has no specific context and therefore chooses to simply represent the indicator with no additional context. Though it's suggested that some context always be given with an indicator if possible, in this case the organization does not have enough additional context to add anything.

Data model: Because this indicator doesn't include any context (see scenario above), the indicator itself is the only top-level component. Within the indicator, the URL is represented as a URI Object with the Type set to "URL" and the Value set to the malicious URL itself (http://x4z9arb.cn/4712/). **[23]**

3.4 Impacts of Malicious URL

Financial fraud, phishing, online gambling, fake TV shopping, fraudulent prize winning, and spam SMS in social networks are all common uses of the internet by criminals [**24**]. The Internet's dark side has emerged and has befuddled the world [**25**]. The detection of malicious Uniform Resource Locators (URLs) is still a problem that causes significant losses each year, particularly in China. In recent years, mainland Chinese citizens have lost more than 20 billion Yuan per year as a result of various forms of phishing, the majority of which were carried out using fake websites located outside of China. Furthermore, the ubiquitous use of

smart phones encourages a rapid increase in mobile and Quick Response (QR) code phishing activities, which encode fake URLs in QR codes to deceive people, particularly seniors. With QR codes being used in almost every aspect of life in China, safety is a major concern for QR code payments. From August 2011 to May 2016, the number of malicious URLs reported to the Anti-phishing Alliance of China (APAC1) was shown in Figure:1. On average, more than 10,000 phishing URLs were reported to APAC per month. The curve at the bottom of Figure 3.1 depicts the trend in the number of malicious websites after adjusting for seasonal effects.[**25,26]**

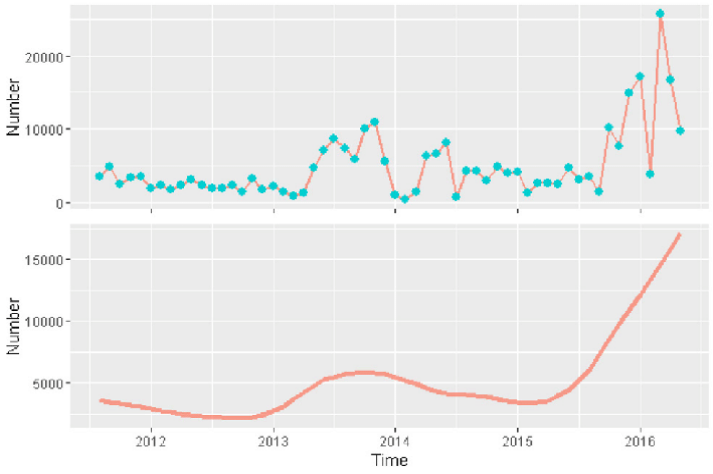

Figure 3.1:Malicious URLs of APAC per month

Moreover, Malware has infected 18.5 million websites, according to [B22]. Furthermore, according to Google's safe browsing report [**B23**] there will be two million phishing websites in September 2020, an increase of nearly 2800% from September 2010.

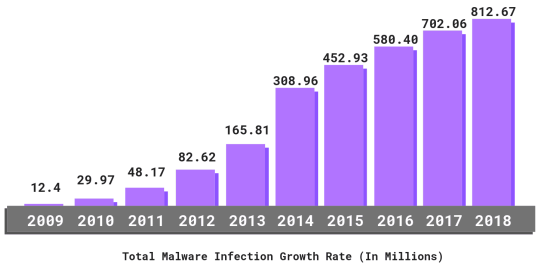

Figure 3.2: Internet Security Threat Report 2019

Also, according to the Figure 3.2. 2019 semantic monthly Internet Security Threat Report 2,**[26]**

- The number of Web Attacks went up by 56% while the number of attack groups using destructive malware rose by 25%. One in ten URLs are malicious.
- In 2018, small organization employees were more likely to fall victim to spam, phishing and email malware as compared to large scale organizations.
- The phishing rate decreased from 1 in 2,995 in 2017 to 1 in 3,207 in 2018 for emails.
- The email spam rate went up to over 54.5% in a month.

CHAPTER FOUR

Feature Representation of Malicious URLs

As previously mentioned, the quality of the training data, which is dependent on the quality of feature representation, is crucial to the performance of a machine learning model. The aim of feature representation is to discover a mapping g: Uϵ R^d such that g(u) $\rightarrow$x where xϵ R^d is a d-dimensional feature vector, which can be input into machine learning models. Given a URL u ϵU, where U is a domain of any valid URL strings. The feature representation procedure can be further divided into the following two steps:

1. Feature Collection: This engineering-focused stage tries to gather significant information regarding the URL. This information includes things like the URLs' inclusion on a blacklist, features extracted from the URL String, information about the host, the website's HTML and JavaScript content, popularity data, etc. **[27]**

2. Feature Pre-processing: The unstructured information about the URL (such as the textual description) is formatted correctly and turned into a numerical vector during the feature pre-processing stage so that it may be used as input by machine learning techniques. Numerical data, for instance, can be used just as is, and bag-of-words is frequently used to represent textual or lexical content.

Researchers have suggested various attributes that can be exploited to offer helpful information for the detection of fake URLs. Blacklist Features, URL-based Lexical Features, Host-based Features, Content-based Features, and Others are the categories into which we divide these features (Context and Popularity).**[27]**

While some are quite instructive, getting these features can be very pricey. Each has its advantages and disadvantages. Similar to this, many features have various pre-processing difficulties and security issues. We will next go into great depth about each of these feature categories.

4.1 Blacklist Features

As previously indicated, using blacklists is an easy way to spot malicious URLs. The list includes any existing URLs that have been determined to be harmful (either by in-depth investigation or crowdsourcing). Blacklisting, however, has been observed to suffer from nontrivial high false negatives **[27]** due to the challenge of keeping exhaustive up-to-date lists, despite its simplicity and ease of implementation. As a result, rather than using blacklist presence as the sole deciding factor, it can be employed as a strong feature. A blacklist's inclusion was specifically employed by **[28]** as a feature by 6 separate blacklist service providers. In comparing the efficiency of these features to other features, they found that blacklist features by themselves did not perform as well as other features, but that when combined with other features, the prediction model performed better overall.

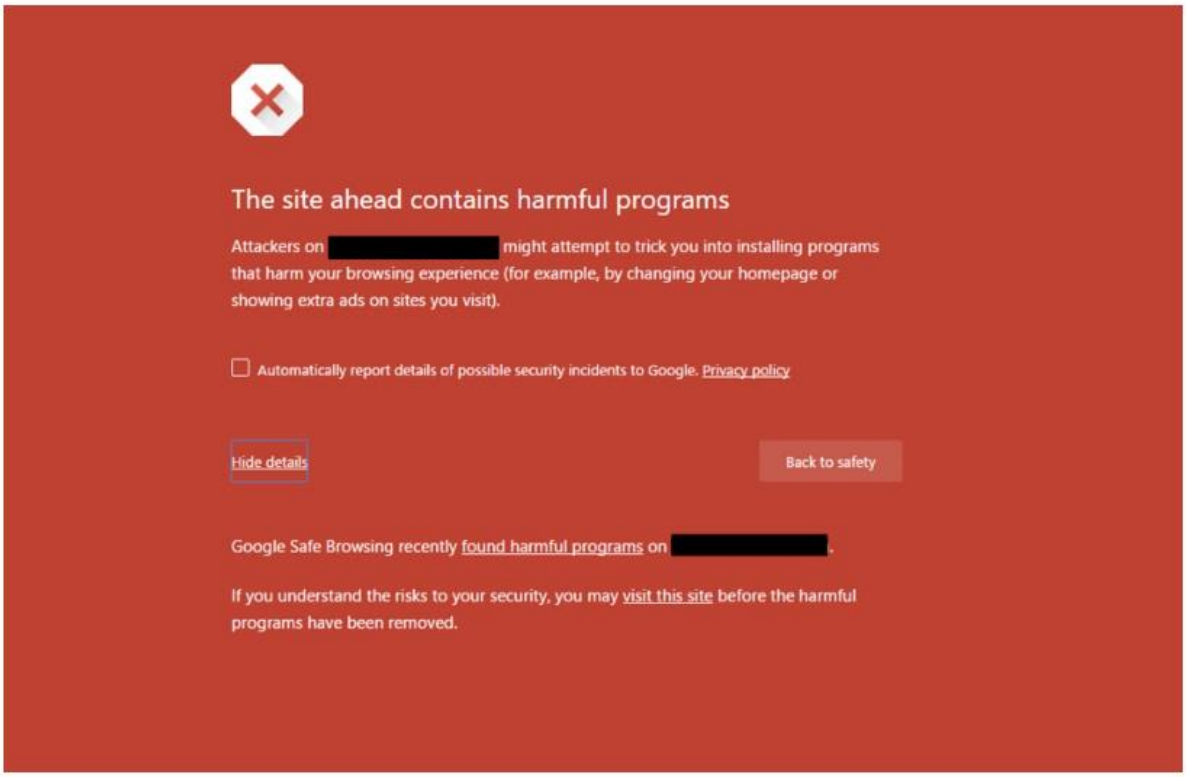

Figure 4.3: Blacklist URL [10]

Noted that many attackers made modest changes to the original URL in order to avoid detection via blacklisting. By using five heuristics—replacing Top-Level Domains (TLDs), IP address equivalence, directory structure similarity, query string substitution, and brand name equivalent—they proposed to expand the blacklist. They also developed a rough matching approach because even a slight mismatch from the blacklist database can allow a

malicious URL to go unnoticed. Blacklist characteristics for machine learning techniques may be derived using similar strategies. Similar methods were used by to generate automated URL blacklists Which created a technique for performing domain blacklisting in advance.

4.2 Lexical Features

Lexical features come from the characteristics of the URL name (or the URL string). The lengths of the URL, the primary domain, the maximum token domain, the average path length, and the average domain token length are some of these features. The reasoning for this is that a URL's maliciousness should be able to be determined based on how it "looks." For instance, several obfuscation techniques attempt to "appear" like safe URLs by imitating their names and making a little change to them. These lexical features are commonly integrated with a variety of additional factors to improve model performance (such as host-based features). It is not possible to use the original URL name directly in machine learning. It is necessary to analyze the URL string in order to extract important characteristics.

The statistical characteristics of the URL string, such as its length and the length of each of its components (Hostname, Top Level Domain, Primary domain, etc.), as well as the number of unusual characters, are among the most frequently utilized lexical aspects and these were to propose word extraction from the URL string was **[29]**. Each segment of the string that was separated by a special character (such as "/", ".", "?", "=", etc.) formed a word. A dictionary was built using all the various word kinds found in all the URLs, making each word a feature. The feature's value would be 1 if the word appeared in the URL and 0 otherwise. This paradigm is frequently referred to as the "bag of words."

When employing the bag-of-words model directly, the information about the words' order within the URL is lost. The full bag-of-word features method can be thought of as a fuzzy blacklist method that is compatible with machine learning. It assigns scores to the URL based on the smaller parts of the URL string rather than focusing on the complete URL string. Although this strategy provides us with a large number of features, it may have issues when complex algorithms are applied to them. Hackers might build harmful URLs algorithmically to avoid being detected by blacklists. Given that algorithmically created URLs may contain terms that have never before been used, using the bag-of-words feature for such URLs is likely to perform poorly.

Table 4. 1: List of URL features in Lexical Feature Group

No	Feature group	Feature	Data type	Description
1		NumDots	numeric	Number of character '.' in URL
2		SubdomainLevel	numeric	Number of subdomain levels
3		PathLevel	numeric	The depth of URL
4		UrlLength	numeric	The length of URL
5		NumDash	numeric	Number of the dash character '-'
6		NumDashInHostname	numeric	Number of dash character in the hostname
7		AtSymbol	boolean	There exists a character '@' in URL
8		TildeSymbol	boolean	There exists a character '~' in URL
9		NumUnderscore	numeric	Number of the underscore character
10		NumPercent	numeric	Number of the character '%'
11		NumQueryComponents	numeric	Number of the query components
12		NumAmpersand	numeric	Number of the character '&'
13		NumHash	numeric	Number of the character '#'
14	**Lexical group**	NumNumericChars	numeric	Number of the numeric character
15		NoHttps	boolean	Check if there exists a HTTPS in website URL
16		IpAddress	boolean	Check if the IP address is used in the hostname of the website URL
17		DomainInSubdomains	boolean	Check if TLD or ccTLD is used as a part of the subdomain in website URL
18		DomainInPaths	boolean	Check if TLD or ccTLD is used in the link of website URL
19		HttpsInHostname	boolean	Check if HTTPS is disordered in the hostname of website URL
20		HostnameLength	numeric	Length of hostname
21		PathLength	numeric	Length of the link path
22		QueryLength	numeric	Length of the query
23		DoubleSlashInPath	boolean	There exists a slash '//' in the link path
24		NumSensitiveWords	numeric	Number of sensitive words (i.e., “secure”, “account”, “webscr”, “login”, “ebayisapi”, “sign in”, “banking”, “confirm”) in website
25		EmbeddedBrandName	boolean	There exists a brand name in the domain
26		PctExtHyperlinks*	float	The percentage of external hyper links in the HTML source code of website

4.3 Host Based features

These characteristics are taken from the host properties of the URLs. These characteristics reveal the whereabouts of malicious servers, their identities, and the extent to which certain host-based characteristics contribute to the maliciousness of the URL. It looked at how a few host-based factors affected how dangerous URLs were. The usage of Short URL services by phishers, the almost quick time-to-live after malicious URLs were registered, and the widespread use of botnets to host themselves on several computers in various nations were some of the important findings. Host-based attributes consequently became crucial in identifying fraudulent URLs.**[29]**

Table 4. 2: List of URL features in Host-Based Feature Group

No	Feature group	Feature	Data type	Description
27		PctExtResourceUrls*	float	Percentage of URL external resource in HTML source codes of website
28		ExtFavicon*	boolean	Check if favicon is installed from a hostname different from the URL hostname of website
29		InsecureForms*	boolean	Check if actions in the form containing the contend of URL without HTTPS protocol
30		RelativeFormAction*	boolean	Check if the action form contains a relative URL
31		ExtFormAction*	boolean	Check if the action form contains an external URL
32		AbnormalFormAction*	boolean	Check if the action form contains an abnormal URL.
33		PctNullSelfRedirectHyperlinks*	float	Percentage of hyperlinks containing an empty value, an auto-redirecting value, such as "#", URL of current website, or some abnormal values such as "file://E:/"
34		FrequentDomainNameMismatch	boolean	Check if the most frequent hostname in the HTML source code does not match the URL of website.
35		FakeLinkInStatusBar*	boolean	Check if HTML source code contains a JavaScript command on MouseOver to display a fake URL in the status bar
36	**Host-based feature group**	RightClickDisabled	boolean	Check if HTML source code contains a JavaScript command to turn off the right click of the mouse
37		PopUpWindow	boolean	Check if HTML source code contains a JavaScript command to start a popup window
38		SubmitInfoToEmail	boolean	Check if HTML source code contains "mailto" in the HTML
39		IframeOrFrame	boolean	Check if iframe or frame is used in HTML source codes
40		MissingTitle	boolean	Check if the title tag is empty in HTML source codes
41		src_eval_cnt	int	Number of function eval () in HTML source codes
42		src_escape_cnt	int	Number of function escape () in HTML source codes
43		src_exec_cnt	int	Number of function exec() in HTML source codes
44		src_search_cnt	int	Number of function search() HTML source codes
45		ImagesOnlyInForm*	boolean	Check if actions in the form of HTML source code does not contain text, but only images
46		rank_country	Boolean	Current country rank of website URL is in top 1 million of Alexa
47		rank_host	Boolean	The rank of the host website URL is in top 1 million of Alexa
48		AgeDomain	int	The age of domain since it is registered

4.4 Content-based Feature

Features that require downloading the complete webpage are referred to as content-based features. These are "heavy-weight" features compared to URL-based features because a lot of data needs to be extracted and there may be security issues at the same time. But it makes sense to believe that having more data accessible on a specific webpage will result in a more accurate prediction model. A more thorough investigation of the content-based features may also aid in the early detection of threats if the URL-based features are unable to identify a malicious URL. The HTML and JavaScript used on a webpage are the main sources of its content-based functionality.**[28]**

4.5 Other Features

Short URL service providers, which enable the original URL to be represented by a shorter string, have expanded during the past several years. This makes it possible to share the URLs on social media sites like Twitter, where the lengthy URLs would not fit under a tweet's 140-character limit. Sadly, this has also grown to be a well-liked obfuscation method for harmful URLs. Despite their best efforts, the short URL service providers struggle to effectively prevent the creation of short URLs for fraudulent users. As a result, a recently active study area has emerged where context-features of the URL, which has been shared, are retrieved.

CHAPTER FIVE

Introduction of Machine Learning Algorithms

Machine learning (ML) is a form of artificial intelligence (AI) that enables software applications to predict outcomes more accurately. Machine algorithms use ancient information as enter for expecting new output values.

Machine learning is commonly used in recommendation engines. Common applications include fraud detection, spam filtering, malware threat detection, business process automation (BPA), and predictive maintenance.

Machine learning is important for businesses because it provides insights into trends in customer behaviour and business operational patterns, as well as assists in developing new products. Many of today's leading companies, including Facebook, Google, and Uber, rely heavily on machine learning. Machine learning has become a significant competitive differentiator for many businesses.

The way an algorithm learns to improve its prediction accuracy is frequently used to classify classical machine learning. There are four types of learning: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. The algorithm that data scientists use is determined by the type of data they want to predict. **[30]**

- *Supervised learning*: In this type of machine learning, data scientists provide labeled training data to algorithms and specify which variables they want the algorithm to look for correlations in. The algorithm's input and output are both given.
- *Unsupervised learning:* In this type of machine learning, algorithms that train on unlabelled data are used. The algorithm searches for meaningful connections in data sets. Predetermined data is used to train algorithms, as are the predictions or recommendations they generate.
- *Semi-supervised learning:* This machine learning approach combines the two preceding types. Although data scientists may provide mostly labeled training data to an algorithm, the model is free to explore the data on its own and develop its understanding of the data set.

- *Reinforcement learning:* Reinforcement learning is typically used by data scientists to teach a machine to complete a multi-step process with clearly defined rules. Data scientists program an algorithm to complete a task and provide it with positive or negative cues as it determines how to complete the task. However, the algorithm generally chooses what actions to take at each stage.

The data scientist must train the algorithm with both labelled inputs and desired outputs in supervised machine learning. The following tasks benefit from supervised learning algorithms:

- *Binary classification:* categorizing data into two groups.
- *Multi-class classification:* Selecting from more than two options.
- *Regression modeling**:*** Predicting continuous values.
- *Ensembling:* Combining the predictions of multiple machine learning models to produce an accurate prediction.

We have to use Supervised Machine Learning algorithms for detecting Malicious URLs. Our classification is Multi-class Classification where we have four types of data. For that, the Algorithms we used to detect Malicious URLs are as follows:

1. Decision Tree classifier
2. Random Forest Classifier
3. K-Nearest Neighbour's (KNN)
4. Gaussian Naïve Bayes (NB)

5.1 The Model

The proposed machine learning-based malicious URL detection system. The machine learning-based malicious URL detection model has two stages: training and detection.

- *Training stage**:*** To detect malicious URLs, both malicious and clean URLs must be collected. The malicious and clean URLs are then correctly labeled before proceeding to attribute extraction. These attributes will be the most useful in determining which URLs are safe and which are dangerous. The specifics of these characteristics will be presented in this paper. Finally, this dataset is divided into two subsets: training data

for machine learning algorithms and testing data for the testing process. If the machine learning model's classification performance is good (high classification accuracy), it will be used in the detection phase.

- *Detection stage:* Each input URL is subjected to the detection phase. First, the URL will be subjected to attribute extraction. Following that, these attributes are fed into the classifier, which determines whether the URL is clean or malicious.

5.2 URL Attributes

The authors**[31]** listed some main attribute groups for extracting features from a URL string, there are three sources of data.

Lexical Features: These are statistical features derived from the literal URL string. For example, the length of the URL string, the number of digits, the number of parameters in the query part, whether the URL is encoded, and so on.

Host-Based Features: These are characteristics of the URL's host-name properties. These contain information about the webpage's host, such as the country of registration, domain name properties, open ports, named servers, connection speed, time to live from registration, and so on.

Content Features: These are obtained from the webpage's downloaded HTML code. These attributes capture the webpage’s structure and the content embedded within it. These will include script tags, embedded objects, executables, hidden elements, and so on.

5.3 Machine Learning Algorithm Selection

Machine learning algorithms have been extensively researched and applied to detect malicious URLs.

5.3.1 Decision Tree classifier

In a decision tree, the algorithm begins at the root node and works its way up to predict the class of a given dataset. This algorithm compares the values of the root attribute with the values of the record (real dataset) attribute and then follows the branch and jumps to the next node based on the comparison.

The algorithm compares the attribute value for the next node to the other sub-nodes and proceeds. It repeats the process until it reaches the leaf node of the tree. The following algorithm can help you better understand the entire process:

Step-1: Begin the tree with the root node, which contains the entire dataset, says S.

Step 2: Find the top attribute in the dataset using the Attribute Selection Measure (ASM).

Step 3: Subdivide the S into subsets containing potential values for the best attributes.

Step 4: Add the best attribute to the decision tree node.

Step 5: Create new decision trees recursively using the subsets of the dataset created in step 3.

Continue this process until you can no longer classify the nodes and refer to the last node as a leaf node. **[32]**

5.3.2 Random Forest Classifier

One of the Random Forest Algorithm's most important features is its ability to handle data sets with both continuous and categorical variables, as in regression and classification. It outperforms other algorithms in classification problems. Figure 5.4 With the example steps

Step 1: In a Random forest, n random records are chosen at random from a data set with k records.

Step 2: For each sample, a unique decision tree is constructed.

Step 3: Every decision tree will generate a result.

Step 4: The final output for classification and regression is based on majority voting or averaging.

Step 5: Find the predictions of each decision tree for new data points and assign the new data points to the category that wins the competition.

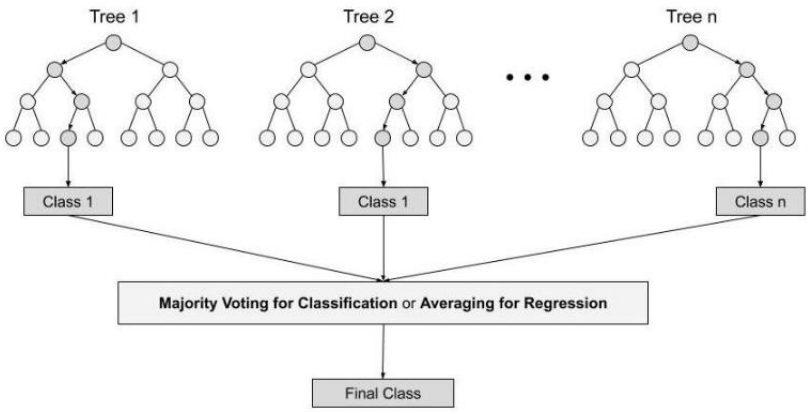

Figure 5.4: Random Forest Tree

Here are some reasons why we should use the Random Forest algorithm:

- It takes less time to train
- It predicts output with high accuracy, and it runs efficiently even with large datasets.
- It can also maintain accuracy when a significant portion of the data is missing. **[33]**

5.3.3 K-Nearest Neighbour's (KNN)

K-Nearest Neighbour's is a straightforward Machine Learning algorithm that employs the Supervised Learning method. The K-NN algorithm assumes that the new and existing cases are similar, assigning the new case to the category that is the most similar to the existing categories. After all of the existing data has been stored, a new data point is classified using the K-NN algorithm based on similarity. This means that new data can be easily classified into a suitable category using the K- NN algorithm. The K-NN algorithm can be used for both regression and classification, but it is most commonly used for classification. Because K-NN is a non-parametric algorithm, it makes no assumptions about the underlying data. It is also known as a lazy learner algorithm because it does not immediately learn from the training set; instead, it stores the dataset and then acts on it during classification. During the training phase, the KNN algorithm simply stores the dataset and classifies it into a similar category to the new data. From Figure 5.5, assume we have two categories, A and B, and a new data point, x1. Determine which of these categories this data point belongs to. To solve this type of problem, a K-NN algorithm is required. We can easily identify the category or class of a particular dataset using K-NN. Consider the diagram below: **[30,34]**

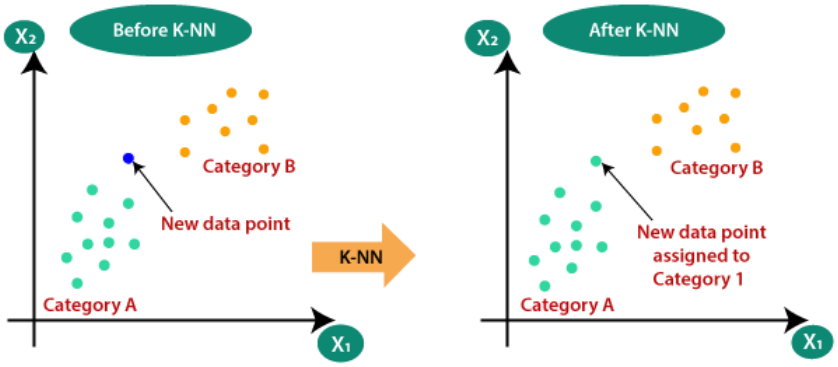

Figure 5.5: K-NN Classification

The K-nearest neighbour's (KNN) algorithm predicts the values of new data points based on 'feature similarity,' which means that the new data point will be assigned a value based on how closely it corresponds to the training set points. The following steps will help us understand how it works.

Step 1: We need a dataset to implement any algorithm. So, in the first step of KNN, we must load both the training and test data.

Step 2: Next, we must select the value of K, i.e. the closest data points. K can be any positive integer.

Step 3 For each point in the test data, do the following:

3.1 Determine the distance between test data and each row of training data using any of the following methods: Euclidean, Manhattan, or Hamming distance. The Euclidean method is the most commonly used method for calculating distance.

3.2 Sort them in ascending order based on the distance value.

3.3 It will then select the top K rows from the sorted array.

3.4 It will now assign a class to the test point based on the most common class of these rows.

Step 4 - Finish. **[34]**

5.3.4 Gaussian Naïve Bayes

The most effective Classification algorithms aid in the development of fast machine learning models capable of making quick predictions. It is a probabilistic classifier, which means it predicts based on an object's probability. Spam filtration, sentiment analysis, and article classification are some popular applications of the Nave Bayes Algorithm.

The Naive Bayes Classifier is a simple Bayes' Theorem:

Bayes' theorem, also known as Bayes' rule or Bayes' law, is a mathematical formula used to calculate the probability of a hypothesis given prior knowledge. It is determined by conditional probability. The formula for Bayes' theorem is as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad \ldots\ldots (5.1)$$

Where,

P(B) = prior probability of B

P(A) = prior probability of class A

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: The probability of the evidence given that the probability of a hypothesis is true.

The Naive Bayes probabilistic classifier generates model rules based on initial knowledge and previous assumptions. It predicts the probability of different classes using a similar attribute. **[35]**

5.4 Evaluation Parameters

***5.4.1 Confusion Matrix*:** The confusion matrix is used to evaluate the classification models' performance for a given set of test data. It denotes a tabular display of Actual vs. Estimated values. **[36]**

True Positive (TP): The predicted value corresponds to the actual value. The actual value was positive, and the model predicted that it would be positive.

False Positive (FP): The predicted value was incorrect. The model predicted a positive value, but the actual value was negative. Also referred to as the Type 1 error.

True Negative (TN): The predicted value corresponds to the actual value. The actual value was negative, and the model predicted that it would be negative.

False Negative (FN): The predicted value was incorrect. The model predicted a negative value, but the actual value was positive. Also referred to as the Type 2 error.

5.4.2 Precision: Precision is defined as the proportion of correctly predicted positive results to all predicted positive results. It assesses the precision of the classifier's output. **[36]**

Precision Score = $\frac{TP}{TP+FP}$ …….(5.2)

5.4.3 Recall: Recall score signifies the model's capability to correctly expect the positives out of actual

positives. It signifies the ratio of true positive to the sum of true positive and false negative. **[B31]**

Recall Score = $\frac{TP}{TP+FN}$ …… (5.3)

5.4.4 Accuracy score: It represents the model's ability to accurately predict both positives and negatives from all predictions, as well as the ratio of the sum of true positives and true negatives from all predictions. **[36]**

Accuracy = $\frac{TP+TN}{TP+FP+TN+FN}$ …… (5.4)

5.4.5 F1- Score: It is the harmonic mean of precision and recall. It is necessary to optimize the system toward either precision or recall, which have a greater impact on the final result**.** **[36]**

F1-Score = $\frac{2*Precision*Recall}{Precision+Recall}$ …… (5.5)

CHAPTER SIX

Introduction of Deep Learning Algorithms

Deep learning is a type of machine learning that is based on artificial neural networks (ANNs) with multiple layers, also known as deep neural networks (DNNs). The concept of deep learning was inspired by the structure and function of the human brain, and the goal is to mimic the way the brain processes information.

Deep learning algorithms are capable of automatically learning features and representations from raw data, without the need for manual feature engineering. Deep-learning architectures such as deep neural networks, deep belief networks, deep reinforcement learning, recurrent neural networks, and convolutional neural networks have been used in fields such as computer vision, speech recognition, natural language processing, machine translation, bioinformatics, drug design, medical image analysis, climate science, material inspection, and board game programs, producing results that are comparable to, and in some cases superior to, traditional methods **[37]**.

In deep learning, the word "deep" refers to the employment of numerous layers in the network. A linear perceptron cannot be a universal classifier, but a network with a non polynomial activation function and one hidden layer of unlimited width can, according to early research. Deep learning is a recent variant that involves an unbounded number of layers of bounded size, allowing for practical application and optimization while maintaining theoretical universality under mild conditions.

We have used Convolutional Neural Network for our deep learning section.

6.1 Framework Overview:

As shown in Figure 6.6, the malicious URL detection system based on neural network is composed of two parts: Training and Testing. In the Training stage, the system passes the labelled data mixed with malicious and normal URLs into the present sandbox, which will automatically open the webpage for screenshot, and generate the website picture of the same size. After the sandbox stage, the image information is converted into the vector matrix so that they can be put into the deep learning model for training purpose. In the testing stage, the testing URL samples need to be pre-processed first. In this case, some URLs are filtered

through the black and white list followed by a simple heuristic feature detection method. And then the remaining URLs will be sent to the sandbox to produce website screenshots. Then, the testing pictures generated from the sandbox are sent to the pretrained CNN model for prediction, and the test results will be divided into two classes, the malicious and benign URLs. **[38]**

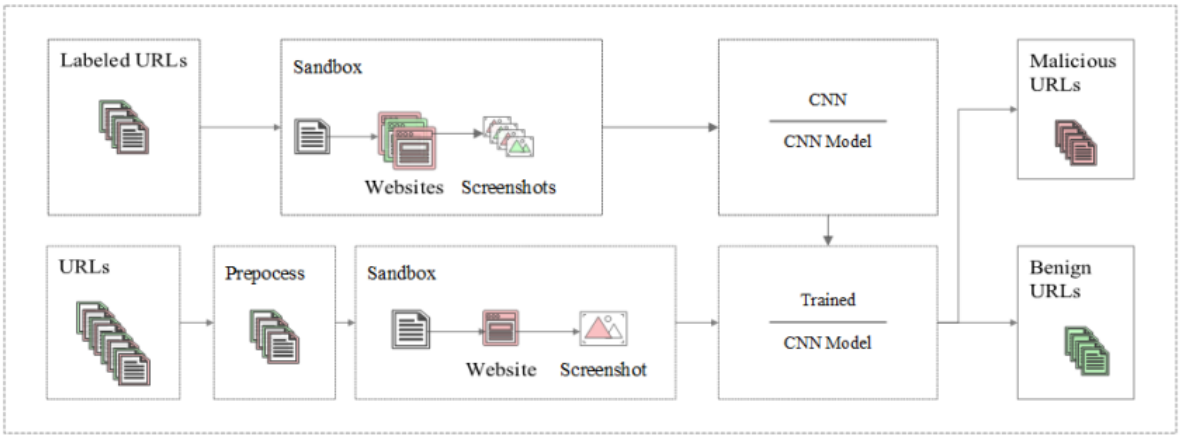

Figure 6.6: Architecture of Malicious URL detection system

6.1.1 Pre-processing Module

In the pre-processing module, We extract the URL's IP address, Whois information, URL length, number of special characters, keywords and other text features, and use simple feature detection algorithms to predict its type. If the predicted result is malicious, the detection result is believed. On the contrary, if the feature prediction result is benign, to reduce the possibility of underreporting, we send it to the sandbox module for web page image capture to prepare for the subsequent deep learning detection method.

6.1.2 Convolutional Neural Network

Convolutional neural network is a multi-layer perceptron designed for two-dimensional or three-dimensional signal recognition based on the human visual nerve mechanism. It maintains high invariance on translation tilt contraction distortion and distortion. Therefore, it is very suitable for recognizing two-dimensional or three-dimensional images that have been deformed to a certain extent, and its application effect in the field of image recognition is very good.

The biggest advantage of CNN over other neural networks is that the neurons in the same feature map share weights, which reduces the number of parameters involved in training and the complexity of the neural network, which makes it easier to converge, and the features

extracted are through training. The data set is automatically learned, avoiding heavy manual feature extraction tasks. A typical convolutional neural network usually contains a two-layer structures: Convolution and Pooling.

In CNN, the input data is processed by a series of convolutional layers, each of which applies a set of filters to the input data. These filters are typically small, square kernels that are designed to detect specific features in the input data. The output of each convolutional layer is then passed through a non-linear activation function, such as a rectified linear unit (ReLU), which is used to introduce non-linearity into the model.

After the convolutional layers, the output is typically passed through one or more pooling layers, which are used to reduce the spatial dimensions of the output. This is done by applying a pooling operation, such as max pooling, which selects the maximum value from a small region of the output. **[37,38]**

Finally, the output of the pooling layers is passed through one or more fully connected layers, which are used to make the final predictions. These layers typically have a large number of neurons and use a sigmoid/softmax activation function to produce probability scores for each class.

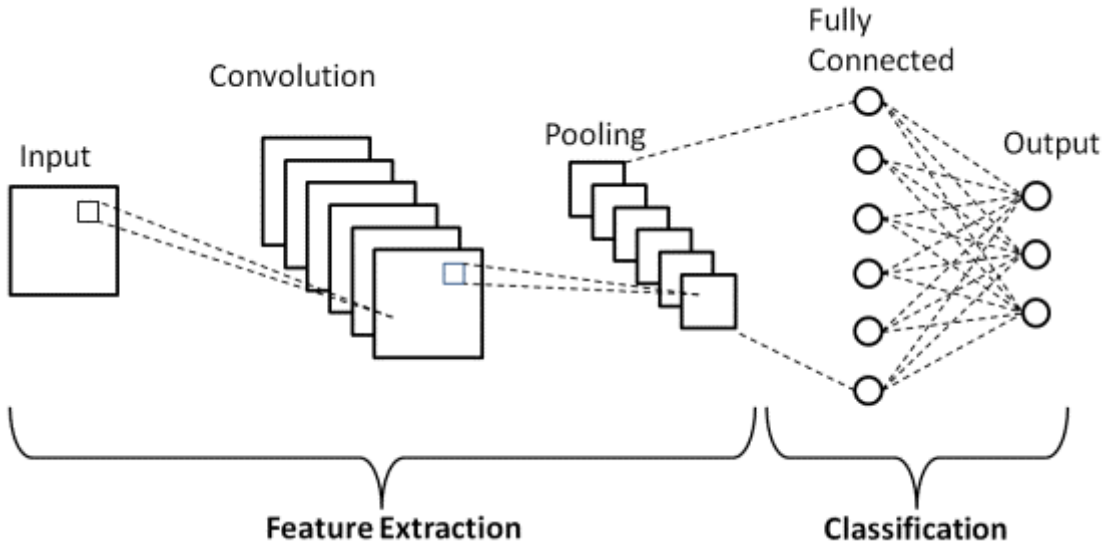

Figure 6.7: The structure of Convolutional Neural Network

Figure 6.7, contain the architecture on Convolutional Neural Network which represent the two layers of CNN model Feature Extraction through the training data and classification.

6.1.3 CNN in Malware Analysis

CNNs have proven their worth in a wide variety of security-related applications. Some of these applications, such as image spam detection are obvious and relatively straightforward applications of CNNs. However, other security domains that do not have any apparent image-based component have also had success with CNNs. By treating executable files as images, researchers have been able to leverage the strengths of CNNs for malware detection, classification, and analysis. For example, many researchers treat executable files as images, and obtain the state-of-the-art result for the malware detection problem. Many of them makes extensive use of transfer learning, whereby the output layer of previously trained CNNs are retrained for the malware detection problem. This results in fast training times and very high malware classification accuracies. So CNN's are successfully applied to a combination of static and dynamic features. [**39]**

CHAPTER SEVEN

Research Methodology

7.1 Proposed Model for Malicious URL

Using Machine Learning models and deep learning models for the detection of Malicious URLs for multiclass classification our proposed model is given below

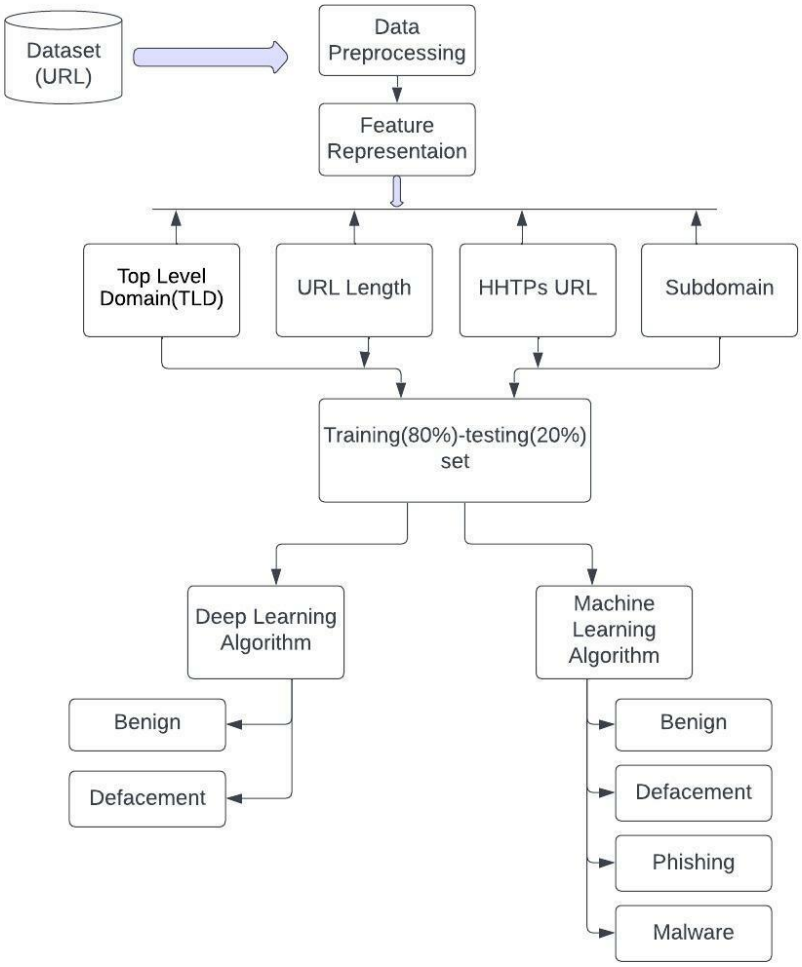

Figure 7.8: Proposed Model for Detection of Malicious URL

7.2 Data visualization

We gathered this dataset to include a large number of Malicious URL examples to build a machine learning-based model to identify malicious URLs. We gathered a massive dataset of 651,191 URLs, of which 428103 were benign or safe URLs, 96457 were defacement URLs, 94111 were phishing URLs, and 32520 were malware URLs.

We used the URL dataset to collect benign, phishing, malware, and defacement URLs (ISCX-URL-2016) We used the Malware domain blacklist dataset to increase the number of phishing and malware URLs. Using the Faizan git repo, we increased the number of benign URLs. Finally, we increased the number of phishing URLs using the Phish tank dataset and the Phish Storm dataset.html) As previously stated, the dataset was compiled from various sources. So, first, we collected URLs from various sources into a separate data frame, then merged them to retain only URLs and their class type.

	url	type
0	br-icloud.com.br	phishing
1	mp3raid.com/music/krizz_kaliko.html	benign
2	bopsecrets.org/rexroth/cr/1.htm	benign
3	http://www.garage-pirenne.be/index.php?option=...	defacement
4	http://adventure-nicaragua.net/index.php?optio...	defacement

Figure 7.9: Data Visualization

Figure 7.9,is showing our dataset which is containing the phishing, benign, defacement and malware data with the URL and type of the URLs. Let's know about the Benign, defacement, malware and phishing URL

Benign: These URLs are safe to visit. Here are some examples of benign URLs**:[B32]**

- mp3raid.com/music/krizz_kaliko.html
- infinitysw.com
- google.co.in
- myspace.com

Defacement: Hackers typically create defacement URLs with the intent of breaking into a web server and replacing the hosted website with one of their own, employing techniques such as code injection, cross-site scripting, and so on. Religious websites, government websites, bank websites, and corporate websites are common targets of URL defacement. Here are some examples of URLs that have been defaced: **[40]**

- http://www.vnic.co/khach-hang.html
- http://www.raci.it/component/user/reset.html

Phishing: Phishing is a type of social engineering attack that is used to steal sensitive user information such as account details and credit card numbers. Hackers attempt to steal sensitive personal or financial information such as login credentials, credit card numbers, internet banking details, and so on by creating phishing URLs. Here are some examples of URLs that have been phishing: **[40][41]**

- roverslands.net
- corporacionrossenditotours.com

Malware: Malware (short for "malicious software") is a file or code that infects, explores, steals, or performs virtually any action desired by the attacker. Here are some examples of Malware URLs: **[40]**

- proplast.co.nz
- http://103.112.226.142:36308/Mozi.m

7.3 Using Machine Learning Models

7.3.1 Data Pre-processing

Data pre-processing includes handling of null values, categories the malware types, extraction of additional features and cleaning the data.

We collected data from Kaggle for a malicious URL dataset which contains approximately 7lakh rows. The dataset contains an equal number of malicious, benign, defacement, and phishing URLs. We have total of 651191 data, where total benign data is 428103, defacement data is 96457, phishing data is 94111 and malware data is 32520. Here, is total datasets is in shown Figure 7.10. The data types included in this dataset are categorical, string and numerical there and no missing values also. Data Sets have two general properties one URL another type. After categorizing drop the string value.

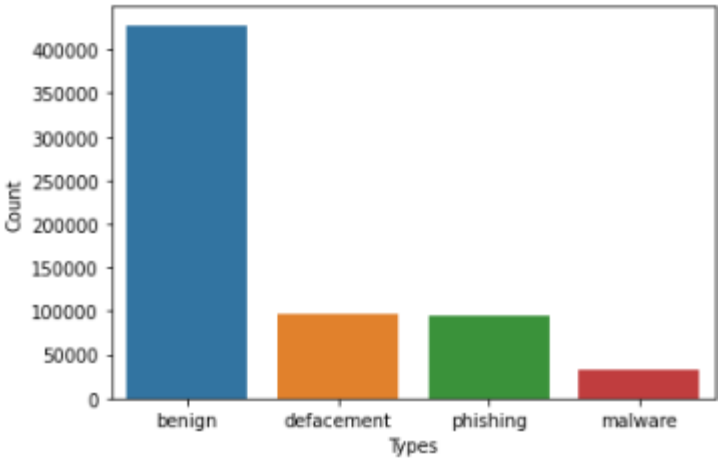

Figure 7.10: Data Barplot

7.3.2 Categoriez data

We have categorized the whole dataset into 4type where we considered benign as number 0, defacement as number1, phishing as number 2 and malware as number3.

	url	type	Category
0	br-icloud.com.br	phishing	2
1	mp3raid.com/music/krizz_kaliko.html	benign	0
2	bopsecrets.org/rexroth/cr/1.htm	benign	0
3	http://garage-pirenne.be/index.php?option=com_...	defacement	1
4	http://adventure-nicaragua.net/index.php?optio...	defacement	1
...	...	...	...
651186	xbox360.ign.com/objects/850/850402.html	phishing	2
651187	games.teamxbox.com/xbox-360/1860/Dead-Space/	phishing	2
651188	gamespot.com/xbox360/action/deadspace/	phishing	2
651189	en.wikipedia.org/wiki/Dead_Space_(video_game)	phishing	2
651190	angelfire.com/goth/devilmaycrytonite/	phishing	2

Figure 7.11: Data Categorization

7.3.3 Feature extraction

It is a process of extracting features from URLs. It is important to generate feature from the URL in order to develop the machine learning model. After generating the features we need to pass these to the machine learning model. Figure 7.12, shows the structure of the URL and the task is to find out the characteristics/features which separate the malicious URLs, phishing URLs, defacement URLs and benign(safe) URLs.

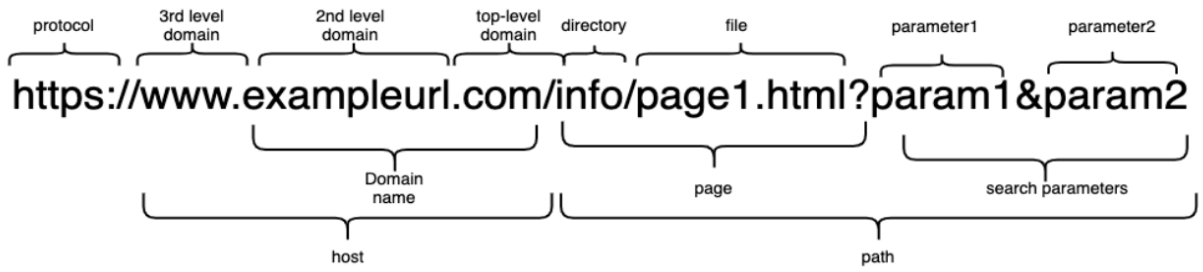

Figure 7.12: URL Feature Extraction

Below is the list of features, we have extracted from the URL and passed onto the machine learning model which then predicts if the URL is malicious, phishing, defacement or benign.

7.3.3.1 URL length count

We counted the URL length because both too long and too short URLs can be suspicious. Malware or phishers may use long URLs to hide the doubtful part in the URL's address bar so that users will click on such URLs without hesitation and fall into traps. They may also shorten the URL to prevent users from seeing the actual contents of the URL or domain name and falling into traps. The average length of benign URLs is found to be 25, and if the URL is longer or shorter than 25, it may be a phishing URL. Here the Figure 7.13, show the length of the URLs.

	url	type	Category	url_len
0	br-icloud.com.br	phishing	2	16
1	mp3raid.com/music/krizz_kaliko.html	benign	0	35
2	bopsecrets.org/rexroth/cr/1.htm	benign	0	31
3	http://garage-pirenne.be/index.php?option=com_...	defacement	1	84
4	http://adventure-nicaragua.net/index.php?optio...	defacement	1	235
...	...	...	...	...
651186	xbox360.ign.com/objects/850/850402.html	phishing	2	39
651187	games.teamxbox.com/xbox-360/1860/Dead-Space/	phishing	2	44
651188	gamespot.com/xbox360/action/deadspace/	phishing	2	38
651189	en.wikipedia.org/wiki/Dead_Space_(video_game)	phishing	2	45
651190	angelfire.com/goth/devilmaycrytonite/	phishing	2	37

Figure 7.13: Data Length Count

7.3.3.2 Extraction of the Top Level Domain(TLD) from URL

In the hierarchical DNS of the Internet, a TLD (top-level domain) is the most generic domain (domain name system). The last part of a domain name is a TLD. In this procedure, the top level domain (TLD) is extracted from the provided URL. In this case, we used the netloc function, which includes the network location, the domain itself, and any subdomains that might be present. That is how we determine the URL's domain.

	url	type	Category	url_len	domain
0	br-icloud.com.br	phishing	2	16	br-icloud.com.br
1	mp3raid.com/music/krizz_kaliko.html	benign	0	35	mp3raid.com
2	bopsecrets.org/rexroth/cr/1.htm	benign	0	31	bopsecrets.org
3	http://garage-pirenne.be/index.php?option=com_...	defacement	1	84	garage-pirenne.be
4	http://adventure-nicaragua.net/index.php?optio...	defacement	1	235	adventure-nicaragua.net
...	...	...	...	...	...
651186	xbox360.ign.com/objects/850/850402.html	phishing	2	39	xbox360.ign.com
651187	games.teamxbox.com/xbox-360/1860/Dead-Space/	phishing	2	44	games.teamxbox.com
651188	gamespot.com/xbox360/action/deadspace/	phishing	2	38	gamespot.com
651189	en.wikipedia.org/wiki/Dead_Space_(video_game)	phishing	2	45	en.wikipedia.org
651190	angelfire.com/goth/devilmaycrytonite/	phishing	2	37	angelfire.com

Figure 7.14: Top level Domain (TLD) Extraction

7.3.3.3 Special Character

We count the number of all special characters (‘@’, ‘?’, ‘-’ , ‘=’ , ‘ .’ , ‘#’, ‘ %’, ‘ +’, ‘ $’, ‘ !’, ‘ *’, ‘ ,’, ‘//’) etc. Excessive use of special characters in URL is suspicious. For example:

Presence of @ symbol in URL: The feature will count a @ symbol if it appears in the URL. The browser frequently skips to the actual address after the "@" symbol when phishers add a specific @ sign to a URL, ignoring everything before it.

Number of dots in Hostname: There are a lot of dots in phishing URLs. For instance, *http://shop.fun.amazon.phishing.com* even though phishing.com is a legitimate domain name, the word "amazon" is used to trick users into clicking on the URL. Three dots are the usual

quantity in safe URLs. The feature will mark a URL as malicious if it contains more than 3 dots.

Prefix or Suffix separated by (-) to domain: It is suspicious if the domain name is separated by the dash (-) symbol. Legitimate URLs rarely use the dash symbol. Phishers include the dash symbol (-) in the domain name to give users the impression that they are visiting a trustworthy website. For instance, the real website address is *http://www.onlineamazon.com*, but phishers can create a fake version of it *http://www.online-amazon.com* to trick innocent users.

URL redirection: If the URL path contains "//," then the feature analyzes it as a defacement url. If the URL path contains the character "//," the user will be forwarded to another website.

Number of slash in URL: The average number of slashes in benign URLs is 5, and if this number is higher then the feature is set the URL as phishing.
In the Figure 15, the extractions of this special characters are shown

	url	type	Category	url_len	domain	@	?	-	=	.	#	%	+	$	!	*	,	//
0	br-icloud.com.br	phishing	2	16	br-icloud.com.br	0	0	1	0	2	0	0	0	0	0	0	0	0
1	mp3raid.com/music/krizz_kaliko.html	benign	0	35	mp3raid.com	0	0	0	0	2	0	0	0	0	0	0	0	0
2	bopsecrets.org/rexroth/cr/1.htm	benign	0	31	bopsecrets.org	0	0	0	0	2	0	0	0	0	0	0	0	0
3	http://garage-pirenne.be/index.php?option=com_...	defacement	1	84	garage-pirenne.be	0	1	1	4	2	0	0	0	0	0	0	0	1
4	http://adventure-nicaragua.net/index.php?optio...	defacement	1	235	adventure-nicaragua.net	0	1	1	3	2	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
651186	xbox360.ign.com/objects/850/850402.html	phishing	2	39	xbox360.ign.com	0	0	0	0	3	0	0	0	0	0	0	0	0
651187	games.teamxbox.com/xbox-360/1860/Dead-Space/	phishing	2	44	games.teamxbox.com	0	0	2	0	2	0	0	0	0	0	0	0	0
651188	gamespot.com/xbox360/action/deadspace/	phishing	2	38	gamespot.com	0	0	0	0	1	0	0	0	0	0	0	0	0
651189	en.wikipedia.org/wiki/Dead_Space_(video_game)	phishing	2	45	en.wikipedia.org	0	0	0	0	2	0	0	0	0	0	0	0	0
651190	angelfire.com/goth/devilmaycrytonite/	phishing	2	37	angelfire.com	0	0	0	0	1	0	0	0	0	0	0	0	0

651191 rows × 20 columns

Figure 7.15: Special Characters Extraction

7.3.3.4 Abnormal URL Count

If a URL is abnormally long or the use of characters, symbols and numbers are too high then the feature will count the URL as an abnormal URL and set it as 1 otherwise 0. In our data, there are approximately 200000 abnormal URLs and nearly 500000 normal URLs which is shown in Figure 7.16.

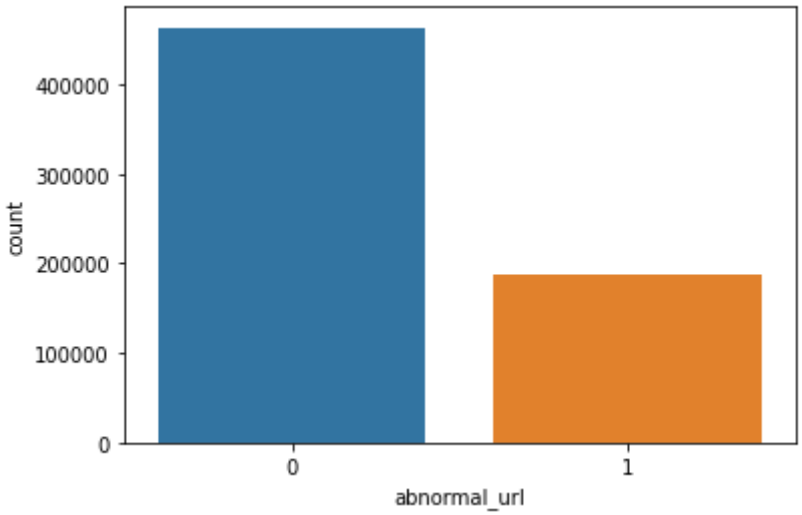

Figure 7.16: Normal URL & Abnormal URL

7.3.3.5 HTTPS Domain URL

The presence of https is critical when checking the URL. If an HTTPS token is present in the URL, the feature is set to 1, otherwise it is set to 0. To mislead users, phishers may simply add the "HTTPS" token to the domain portion of a URL. For instance, *http://www-paypal-it-mpp-home.soft-hair.com.* To trap users, phishers may place HTTPS in front of HTTP, as in the example above. From our total dataset the total HTTPs URL are shown in the Figure 7.17.

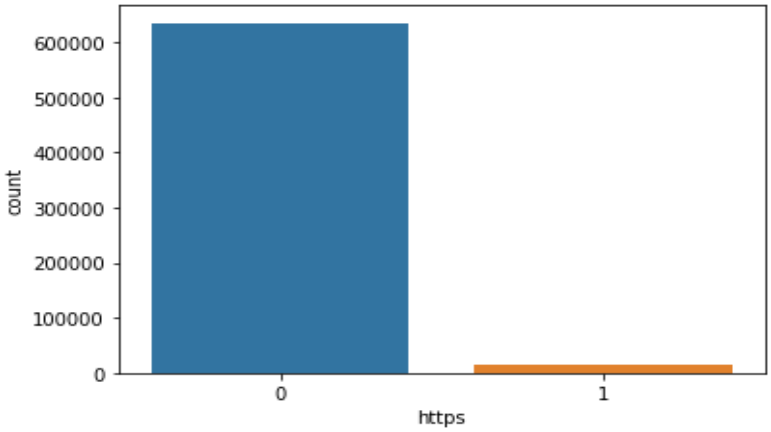

Figure 7.17: HTTPs URLs

7.3.3.6 Presence of IP address in URL

If the IP address is present in the URL, the feature is set to 1, otherwise it is set to 0. The majority of safe sites do not use an IP address as a URL to download a webpage. The presence of an IP address in a URL indicates that the attacker is attempting to steal sensitive information. There are 12488 URLs in our dataset that have IP addresses.

7.3.3.7 Count_digits

The presence of digits in URL generally indicate suspicious URLs. Safe URLs generally do not have digits so counting the number of digits in URL is an important feature for detecting malicious URLs.

7.3.3.8 Count_letters

The number of letters in the URL also plays a significant role in identifying malicious URLs. As attackers try to increase the length of the URL to hide the domain name and this is generally done by increasing the number of letters and digits in the URL.

After applying the feature extractions all Features

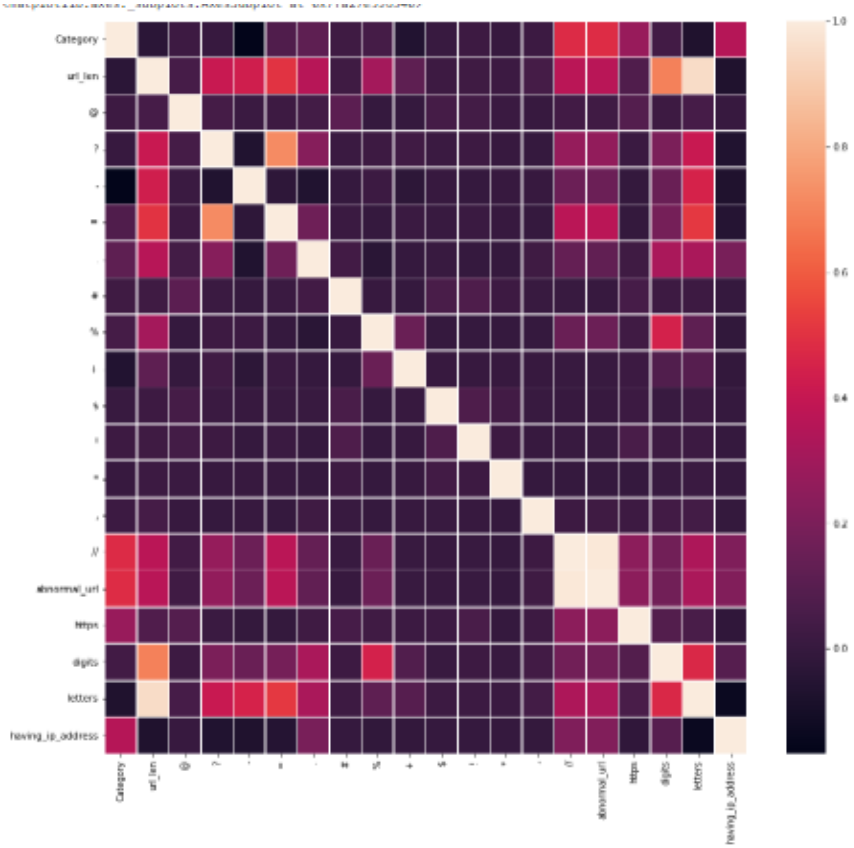

Figure 7.18: Dependent Variables of Feature Extractions

After feature extraction the dependent variables of our dataset are in a confusion matrix which is shown in the Figure 7.18. These variables are used as our input data for the detection of malicious URL.

7.3.4 Data cleaning

In the data cleaning process, clean all character types data and by converting this into a digital number set the data set by using feature extraction as these features will be used as the input features for training the machine learning model.

- Lowercase all the URLs
- Remove any characters that is not in a-z OR A-Z alphabet, that includes generic text, numbers, symbols etc.
- Remove whitespaces
- Remove tabs

So, In ‘X’ we set all values for URLs and in ’y’ set the categories of the dataset.

7.3.5 Splitting Training and Testing data

To evaluate the models in the dataset, we divide it into two parts: one for training and one for testing. We comparing model results using three types of training and testing datasets.

In order to, train the machine learning models, To begin, we use 50% of the dataset for training and 50% for testing. Second, we use 80% of the dataset for training and 20% for testing, and finally, we use 90% of the dataset for training and 10% for testing.

The training set contains approximately 520953 data points, while the testing set contains approximately 130239 data points. Tuning hyperparameters improves accuracy, precision, and recall.

7.4 Using Deep Learning Models

7.4.1 Data Processing

The data we used in this experiment was drawn from URLs (ISCX-URL-2016). At first there was 4 types of data where 651,191 URLs, out of which 428103 benign or safe URLs, 96457 defacement URLs, 94111 phishing URLs, and 32520 malware URLs. Basically, 66% URL is benign or safe URL and other 34% URL is malicious URL. After data cleansing, a total of 651,191 URL samples were used in this experiment, of which 428103 were benign or safe URLs, 96457 were defacement URLs, 94111 were phishing URLs, and 32520 were malware URLs. We classified our data into two sectors such as all the safe URLs like benign, defacement URLs as Benign and other phishing, malware URL as Malicious URLs. We have categorized benign data as 0 and malicious data as 1. In pre-processing we tokenized the data by encoding the token convert these into numerical values after tokenization pad the encoded tokens to the maximum length using a specific symbol or value. This can be done using the pad_sequences function from the Keras library in python, this result used as input feature of deep learning model.

7.4.2 Feature Engineering

The characters or tokens in the URL string would probably serve as the feature representation of URLs in a CNN algorithm for malicious URL detection using deep learning. These features are extracted by passing the URL string through convolutional and pooling layers. To determine whether the URL is malicious or not, the extracted features are then sent through fully connected layers. The feature representation may also contain extra details like the domain name, the presence of specific keywords or patterns, and the URL structure.

7.4.2.1 Extraction of 'Domain', 'Subdomain', 'Domain_Sriffix'

The following steps can be used to extract the "domain," "subdomain," and "domain suffix" from a URL for malicious URL detection:

1. To separate the scheme, network location, path, and query parameters from the URL, use the "URL split" function from the Python "urllib. parse" library.
2. Take the output of step 1 and extract the network location, which contains the domain and possibly the subdomain.
3. To match the domain within the network location, use regular expressions.
4. Take the match you got in step 3 and extract the domain from it.
5. Use regular expressions to extract the subdomain and domain suffix from the domain you obtained in step 4 or the Python "tldextract" library to separate the subdomain, domain, and suffix of the given URL.

The extracted 'domain', 'subdomain' and 'domain suffix' will be used as input features to the fully connected layers of the CNN model, where it can be used in combination with other extracted features to make a prediction on whether the URL is malicious or not.

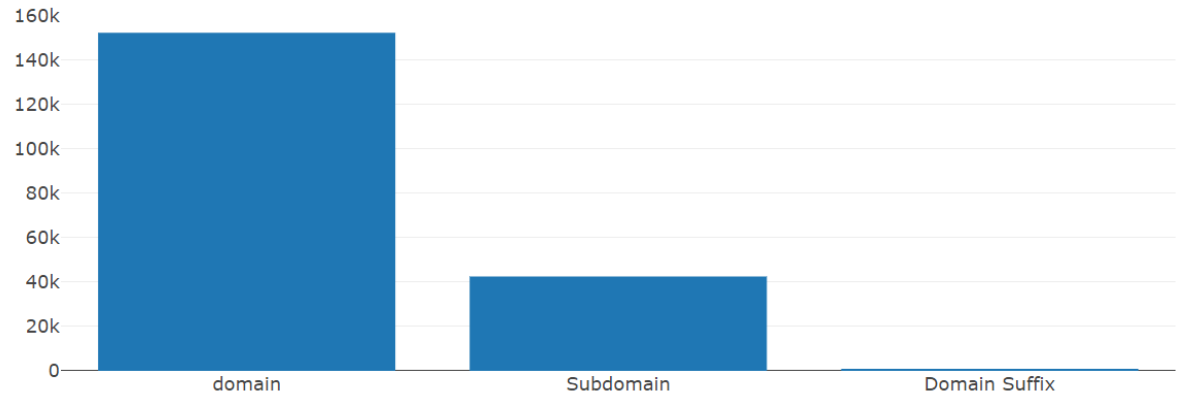

Figure 7.19: Extraction of Domain, Subdomain, Domain Suffix

7.4.2.2 Bag of Words

Bag of Words (BoW) is an algorithm, which is a way of extracting features from text or document and calculating the number of times words appear in the text or document without following any grammatical rule or words order. It is a matrix of tokens in which if the word appears in the text or document, it will give one value to that word otherwise it will give zero.

7.4.2.3 TF-IDF

TF-IDF stands for Term Frequency - Inverse Document Frequency, is a numerical statistic that reflects how important a word is to a document in a collection or corpus. It measures relevance of the word rather than the frequency of the word.

$$\text{Term Frequency, TF} = \frac{\text{Number of occurnce of word in a documentTotal}}{\text{number of words in a document}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (7.6)$$

$$\text{Inverse Document Frequency, IDF} = \log\left(\frac{\text{Total number of documentsNumber}}{\text{of documents containing the word}}\right) \quad \ldots\ldots\ldots\ldots\ldots \quad (7.7)$$

TF-IDF of a word is calculated by the multiplication of Term Frequency score and Inverse Document Frequency score.

If we compare TF-IDF model with Bag of Words we can see, Bag of Words contains only zeros & ones. It gives all words have the same importance and doesn't preserve any semantic information. On the other hand, TF-IDF values a word based on its importance in the whole document or corpus. For feature extraction we have applied TF-IDF in our research work.

CHAPTER EIGHT

Result and Analysis

The scikit-learn tool has been used to import Machine learning algorithms. Each classifier is trained using training set and testing set to evaluate classifiers' performance. The performance of classifiers has been evaluated by calculating the classifier's accuracy score, precision, recall & F1 score.

8.1 Using Machine Learning Models

Table 8. 1: Malicious URL Detection Accuracy using Machine Learning Models

Dataset Split Ratio	ML Classifier	Types	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
80:20	Decision Tree	0	90.89%	92%	97%	94%
		1		93%	96%	94%
		2		80%	56%	66%
		3		94%	91%	92%
	Random Forest	0	91.44%	92%	98%	95%
		1		93%	96%	95%
		2		83%	57%	68%
		3		96%	91%	93%
	K-Nearest Neighbors	0	89.01%	91%	96%	93%
		1		89%	95%	92%
		2		74%	53%	61%
		3		94%	87%	91%
	Gaussian Naïve Bayes	0	78.92%	85%	92%	88%
		1		66%	100%	79%
		2		58%	02%	03%
		3		69%	71%	65%

We see that from table 8.3, the Random forest classifier gives higher accuracy than other classifiers which is 91.44% with a higher rate for Benign, defacement, phishing, and malware. So, Random Forest classifier is best for our dataset.

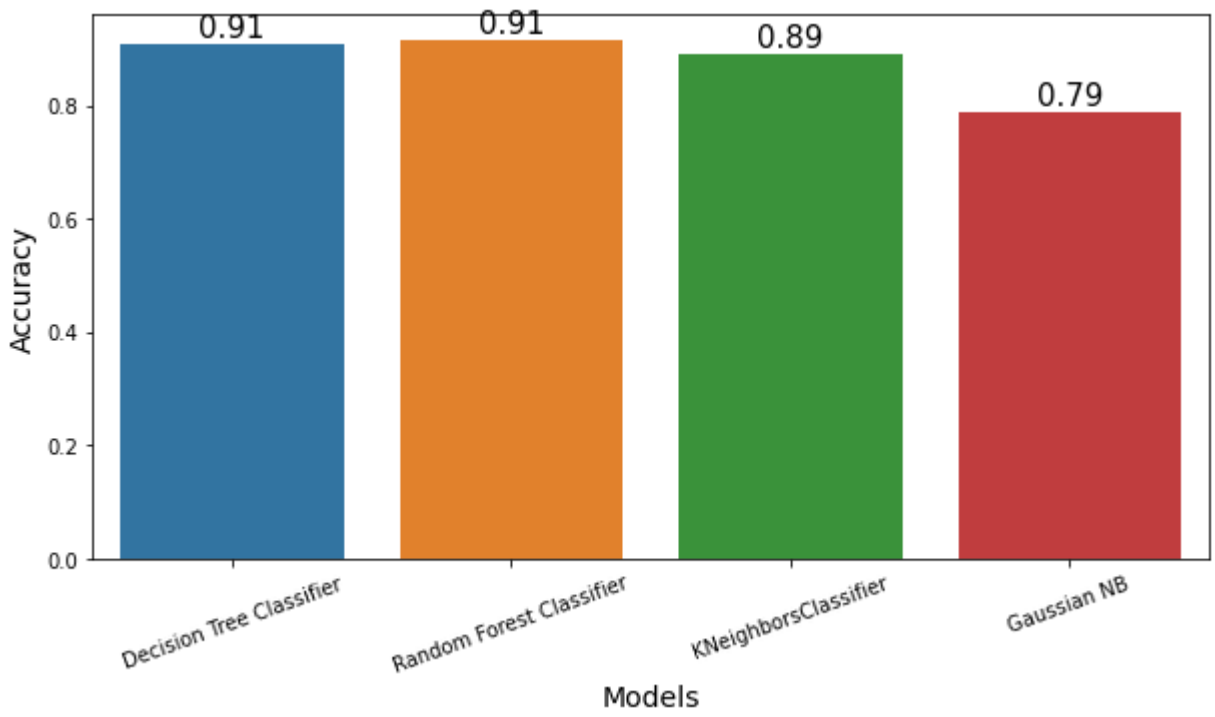

Figure 8.20: ML Classifiers Accuracy

Figure 8.20 is shown that the accuracy of four classifier of Machine learning model.

8.1.1 Applying The Confusion matrix of ML Classifiers

We applied confusion matrix for machine learning models such as Decision tree, Random Forest, K-Nearest Neighbors, Gaussian Naïve Bayes to observe the performance of models

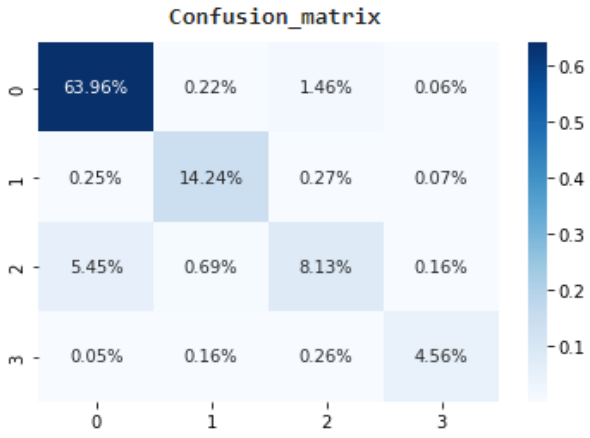

Figure 8.21: Confusion Matrix of Decision Tree

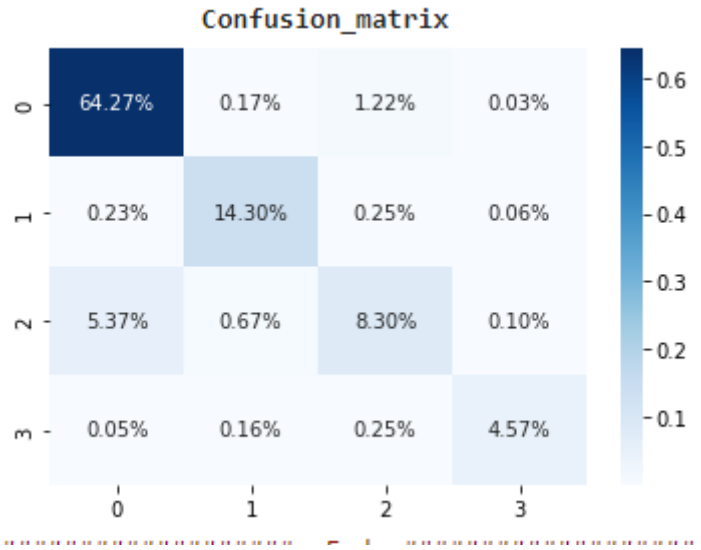

Figure 8.22: Confusion Matrix of Random Forest

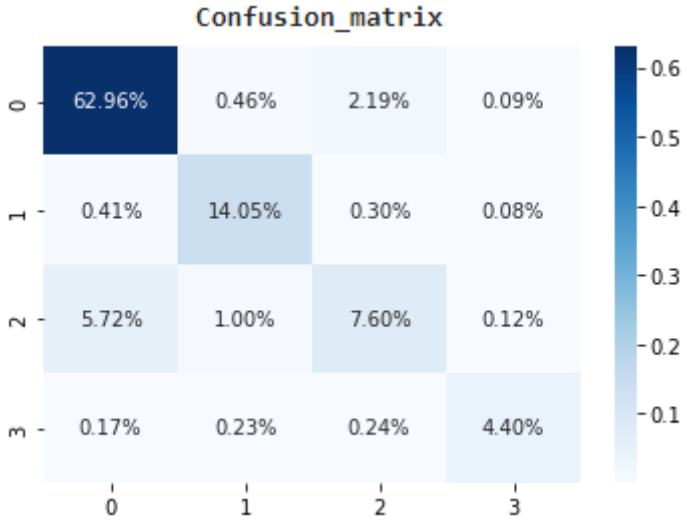

Figure 8.23: Confusion matrix of K-Nearest Neighbour's

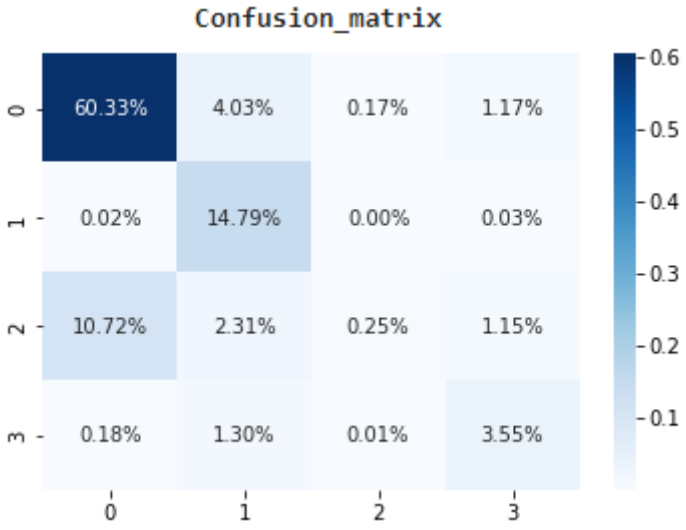

Figure 8.24: Confusion matrix of Gaussian Naïve Bayes

The confusion matrixes show that the model correctly predicts a good number of benign, defacement, phishing and malware URLs. So, based on this figure we have created acceptable machine-learning models.

8.1.2 Comparing Accuracy of ML Classifiers in Different Ratios of the dataset

Table 8. 2: The Comparison on different ratio

Dataset Split Ratio	ML Classifier	Accuracy (%)
50:50	Decision Tree	90.33%
	Random Forest	91.07%
	K-Nearest Neighbors	88.37%
	Gaussian Naïve Bayes	78.99%
80:20	Decision Tree	90.89%
	Random Forest	91.44%
	K-Nearest Neighbors	89.01%
	Gaussian Naïve Bayes	78.92%
90:10	Decision Tree	91.05%
	Random Forest	91.63%
	K-Nearest Neighbors	89.22%
	Gaussian Naïve Bayes	79.05%

Result shows that for every ratio, Random forest algorithm gives better detection accuracy than decision tree, k-Nearest Neighbors and Gaussian Naive bayes algorithms. Result also

shows that detection accuracy of malicious URL increases as more dataset used as training dataset.

After applying into three ratios in the Table 8.2, we achieved best results in 90:10 ration train test set and in three case we achieved the highest accuracy in Random Forest classifier with the less time. All classifiers perform well when 90% of data used as training dataset.

8.2 Using Deep Learning Models

We apply Convolutional Neural Network (CNN) to our dataset. We split the dataset as 80% of training set and 20% of testing set.

We select vocab_size =6800, embedding dimension is 64, that means we are converting every single token of words into 64 dimension and the input length is 234. We have total trainable parameters are 5,484,285. Then, To avoid vanishing gradient problem, we use relu based activation function and in the output layer used the Sigmoid activation function to have better accuracy. After 5epoch of the CNN model's the results are given Figure 8.25 to Figure 8.28.

Layer (type)	Output Shape	Param #
url_input (InputLayer)	[(None, 234)]	0
embedding (Embedding)	(None, 234, 32)	10656
conv1d (Conv1D)	(None, 234, 64)	6208
conv1d_1 (Conv1D)	(None, 234, 64)	10304
subdomain_input (InputLayer)	[(None, 1)]	0
domain_input (InputLayer)	[(None, 1)]	0
domain_suffix_input (InputLayer)	[(None, 1)]	0
concatenate (Concatenate)	(None, 234, 160)	0
embedding_1 (Embedding)	(None, 1, 4)	143028
embedding_2 (Embedding)	(None, 1, 4)	516652
embedding_3 (Embedding)	(None, 1, 4)	3324
flatten (Flatten)	(None, 37440)	0
reshape (Reshape)	(None, 4)	0
reshape_1 (Reshape)	(None, 4)	0
reshape_2 (Reshape)	(None, 4)	0
concatenate_1 (Concatenate)	(None, 37452)	0
dropout (Dropout)	(None, 37452)	0
dense (Dense)	(None, 128)	4793984
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129

Total params: 5,484,285
Trainable params: 5,484,285
Non-trainable params: 0

Figure 8.25: Models Feature Extraction

```
Epoch 1/5
6512/6512 [==============================] - 63s 9ms/step - loss: 0.1141 - precision: 0.9559 - recall: 0.9268 - val_loss: 0.0693 - val_precision: 0.9727 - val_recall: 0.9585
Epoch 2/5
6512/6512 [==============================] - 60s 9ms/step - loss: 0.0669 - precision: 0.9747 - recall: 0.9588 - val_loss: 0.0673 - val_precision: 0.9637 - val_recall: 0.9725
Epoch 3/5
6512/6512 [==============================] - 60s 9ms/step - loss: 0.0538 - precision: 0.9801 - recall: 0.9662 - val_loss: 0.0596 - val_precision: 0.9818 - val_recall: 0.9614
Epoch 4/5
6512/6512 [==============================] - 60s 9ms/step - loss: 0.0450 - precision: 0.9836 - recall: 0.9719 - val_loss: 0.0576 - val_precision: 0.9843 - val_recall: 0.9621
Epoch 5/5
6512/6512 [==============================] - 60s 9ms/step - loss: 0.0395 - precision: 0.9861 - recall: 0.9756 - val_loss: 0.0539 - val_precision: 0.9800 - val_recall: 0.9700
```

Figure 8.26: CNN Model Prediction

The diagram of loss, precision and recall of the dataset for the model

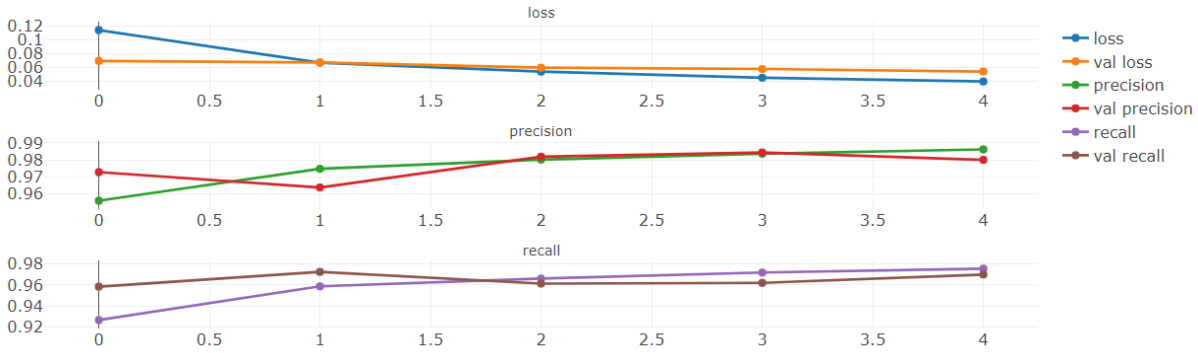

Figure 8.27: Prediction Diagram of CNN Model

Malicious URL detection using CNN model the Accuracy of this model is 98% with 98% of precision, 99% of recall and 99% of f1-score for Benign data (0) and 98% of precision, 97% of recall and 97% of f1-score for Malware data (1) which are shown in the Figure 27.

So, we can say that our model is good classifier through the detection of Malicious URL.

Classification Report:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	85621
1	0.98	0.97	0.97	44618
accuracy			0.98	130239
macro avg	0.98	0.98	0.98	130239
weighted avg	0.98	0.98	0.98	130239

Figure 8.28: Classification Report of CNN model

In the case of ratio 80:20, From Machine Learning and Deep Learning we get our accuracy to 90%. In Machine learning the best accuracy result given by the Random Forest Classifier which is 91.44% on the other hand in deep learning CNN given 98% accuracy result. So in Deep Learning model we get the best accuracy because of in Deep Learning model the dataset trained by the label wise where Random forest classifier predict the output by averaging or majority voting the predictions of all the trees.

CHPATER NINE

Machine learning vs Deep learning

Machine learning and deep learning are two branches of artificial intelligence that are often used interchangeably, but they are not the same thing. Machine learning and deep learning have applications in different area like cyber security[41][45][46][47], medical[42], Natural Language Processing[43][44][48][49].

Machine learning is a broader field that encompasses a wide range of algorithms and techniques for building models that can learn from data. These models can be used for tasks such as prediction, classification, and clustering. Machine learning algorithms can be divided into two main categories: supervised and unsupervised. Supervised algorithms learn from labelled data, while unsupervised algorithms learn from unlabelled data.

Deep learning, on the other hand, is a specific type of machine learning that is based on artificial neural networks. These networks are inspired by the structure and function of the human brain and are composed of multiple layers of interconnected nodes, or "neurons." Deep learning algorithms are particularly well-suited for tasks such as image and speech recognition, natural language processing, and video analysis.

One of the key differences between machine learning and deep learning is the amount of data and computational power required. Deep learning models are highly complex and require large amounts of data and powerful hardware to train. In contrast, traditional machine learning algorithms can often be trained on much smaller datasets and with less powerful hardware.

Another difference is the ability of deep learning to automatically learn features from data, whereas traditional machine learning models rely on manually-engineered features. This makes deep learning particularly useful for tasks where the data is unstructured, such as image and speech recognition, as it can automatically extract useful information from the data.

In summary, machine learning is a broader field that encompasses a wide range of algorithms and techniques, while deep learning is a specific type of machine learning that is based on artificial neural networks and is particularly well-suited for tasks such as image and speech

recognition. Both have their own strengths and weaknesses, and the choice between the two often depends on the specific problem and the available resources.

9.1 Comparison between Machine Learning and Deep Learning Model

We used both machine learning and deep learning techniques to detect malicious URL for our thesis article. With the help of machine learning algorithms, we were able to achieve relatively high test accuracies, especially with Random Forest Classifier which had accuracy values of 91.43% and higher levels of precision, recall, and f-1 score. However, using deep learning on these datasets improves test accuracy compared to machine learning, especially when using Convolutional Neural Network (CNN), which has Validation Accuracy of 98%.

Therefore, if we compare these results based on our findings on these specific datasets, we may draw the conclusion that deep learning algorithm which means CNN architecture represent the best methods for Malicious URL Detection.

CHAPTER TEN

Conclusion

10.1 Research Challenges

Malicious URL detection using machine learning and deep learning is an active area of research that has the potential to improve the security of online systems and protect users from cyber attacks. However, there are several challenges that researchers must overcome in order to develop effective and reliable systems.

There are a lot of challenges we have to face during the research. As our dataset is divided into 4 classes which are benign, defacement, phishing, and malware so we catagorise the classes as 0,1,2 and respectively. For that reason, we have to face some challenges at the time of applying K-Nearest Neighbors classifier as it takes more time to evaluate model.

Moreover , we have to face challenges while using Deep learning algorithm. As we have four classes in our dataset, it was impossible to get the output. The main reason of this problem is we have use sigmoid function in deep learning algorithm. We now that sigmoid function is used for binary classification method where only two classes exist. So, we need to catagorize the dataset into 2 classes and after that we apply the CNN algorithm .

Another challenge is the high dimensionality of the data. URL contain a large amount of information, and extracting relevant features that can be used for detection is difficult. Additionally, the large number of features make it difficult to train machine learning models and overfitting occurs. Thats why we need to extract some features.

10.2 Future Work

Malicious URL detection using machine learning and deep learning is an active area of research that has the potential to improve the security of online systems and protect users from cyber attacks. Thats why we will continue our research work in future, though there are several challenges that we need to overcome to develop effective systems.

One area of future work is in the development of more advanced machine learning and deep learning models. With the increasing power of these techniques, it is likely that more complex and sophisticated models will be developed that can better handle the high dimensionality of the data and adapt to the evolving nature of cyber threats. Thats why we

are interested in applying LSTM, Bi LSTM, Google-net algorithm on our datasets to see how it performs.

Moreover, our model's evaluation rate is 91.44% in the Random forest classifier. But we want to increase our accuracy rate. For that, in future we will apply for XGBoost Classifier, Light GBM classifier and SVM classifier for better results. SVM classifier gives best accuracy in all cases of dataset.

10.3 Conclusion

An essential component of online security is the detection of malicious URLs since it protects users from malware attack and online attacks. Malicious URL detection systems are becoming more accurate and effective due to machine learning and deep learning. These methods enable the detection system to develop and improve over time by learning from instances of both good and bad URLs. In this paper, we aimed to find the best-performing model using our URL-based features in a multiclass ensemble classification setting. We have proposed an approach to detect malicious URLs and tested our approach applying machine learning algorithms such as Gaussian Naive Bayes, Random Forrest, k-Nearest Neighbors, Decision Tree and deep leaning algorithms such as CNN. We used various machine learning model and deep learning model for our detection and found the best one. In all machine learning algorithms, Random Forest Classifier provides better accuracy which is 91.43%. We have applied the Confusion Matrix for each of these algorithms to observe how well we build the model. In this case, we have achieved high precise, recall and f-1 score. For applying Deep learning algorithm we catagorize the dataset as binary dataset and it provides better accuracy than machine learning algorithms which provides accuracy of 98%. We also highlight some recent URL attack data all over the world. Moreover, in our paper we gave a clear idea about the threatening position of Malicious URL attack and its detection techniques using machine learning and deep learning.

References

1. Patgiri, R., Katari, H., Kumar, R., Sharma, D. (2019). Empirical Study on Malicious URL Detection Using Machine Learning. In: Fahrnberger, G., Gopinathan, S., Parida, L. (eds) Distributed Computing and Internet Technology. ICDCIT 2019. Lecture Notes in Computer Science(), vol 11319. Springer, Cham.
2. Eshete, A. Villafiorita and K. Weldemariam, "Malicious Website Detection: Effectiveness and Efficiency Issues," 2011 First SysSec Workshop, Amsterdam, Netherlands, 2011, pp. 123-126, doi: 10.1109/SysSec.2011.9
3. Blackbarry. (n.d.). lackBerry Uncovers Massive Hack-For-Hire Group.
4. Atleson, M. (2019, July 29). Equifax Data Breach: Beware of Fake Settlement Websites. Retrieved from U.S. Federal Trade Commission.
5. Blackbarry. (n.d.). lackBerry Uncovers Massive Hack-For-Hire Group. “COVID-19, Info Stealer and the Map of Threats,” Reason Labs
6. Eshete, B.; Villafiorita, A.; Weldemariam, K. BINSPECT: Holistic Analysis and Detection of Malicious Web Pages. In Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering; Springer: Berlin, Germany, 2013; Volume 106 LNICS, pp. 149–166.
7. Ma, J.; Saul, L.K.; Savage, S.; Voelker, G.M. Beyond blacklists: Learning to detect malicious web sites from suspicious URLs. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 28 June–1 July 2009; pp. 1245–1253.
8. Curtsinger, C.; Livshits, B.; Zorn, B.; Seifert, C. ZOZZLE: Fast and precise in-browser JavaScript Malware detection. In Proceed-ings of the 20th USENIX Security Symposium; USENIX Association, San Francisco, CA, USA, 8–12 August 2011; pp. 33–48.
9. Abdelhamid, N.; Ayesh, A.; Thabtah, F. Phishing detection based Associative Classification data mining. Expert Syst. Appl. 2014, 41, 5948–5959.
10. Jeeva, S.C.; Rajsingh, E.B. Intelligent phishing url detection using association rule mining. Hum. Centric Comput. Inf. Sci. 2016, 6, doi:10.1186/s13673-016-0064-3.
11. Kim, S.; Kim, J.; Nam, S.; Kim, D. WebMon: ML- and YARA-based malicious webpage detection. Comput. Networks 2018, 137, 119–131, doi:10.1016/j.comnet.2018.03.006.

12. Li, Y.; Yang, Z.; Chen, X.; Yuan, H.; Liu, W. A stacking model using URL and HTML features for phishing webpage detection. Futur. Gener. Comput. Syst. 2019, 94, 27–39, doi:10.1016/j.future.2018.11.004.
13. Google Safe Browsing. Available online: https://safebrowsing.google.com/ (accessed on 20 November 2019).
14. Cao, Y.; Han, W.; Le, Y. Anti-phishing based on automated individual white-list. In Proceedings of the ACM Conference on Computer and Communications Security, Alexandria VA, USA, 27–31 October 2008; pp. 51–59.
15. Liu, Y., & Zhang, M. (November 2012). Financial websites oriented heuristic anti-phishing research. Proceedings of the 2012 IEEE 2nd international conference on cloud computing and intelligence systems. IEEE, Hangzhou, China, vol. 2, pp. 614–618.
16. N. M. Shekokar, C. Shah, M. Mahajan, and S. Rachh, "An ideal approach for detection and prevention of phishing attacks," *Procedia Computer Science*, vol. 49, pp. 82–91, 2015.
17. Dharmaraj Rajaram Patil and JB Patil. 2015. Survey on Malicious Web Pages Detection Techniques. International Journal of u-and e-Service, Science and Technology (2015).
18. Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. 2009. Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining.
19. Bin Liang, Jianjun Huang, Fang Liu, Dawei Wang, Daxiang Dong, and Zhaohui Liang. 2009. Malicious Web Pages Detection Based on Abnormal Visibility Recognition. In EBISS. IEEE.
20. Serena Raymond, "Webshrinker's malicious URL categories", 2020
21. AO Kaspersky," What is cybercrime? How to protect yourself from cybercrime",2022
22. F. Vanhoenshoven, G. N´apoles, R. Falcon, K. Vanhoof and M. K¨oppen, *Detecting malicious URLs using machine learning techniques*, 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8, doi: 10.1109/SSCI.2016.7850079.
23. The OASIS Cyber Threat Intelligence (CTI) TC supports automated information-2017

24. . S. Jeong, J. Lee, J. Park, C. Kim, The social relation key: A new paradigm for security, Inf. Syst. 71 (2017) 68–77.
25. Won Kim, Ok-Ran Jeong, Chulyun Kim, Jungmin So, The dark side of theinternet: Attacks, costs and responses, Inf. Syst. 36 (2011) 675–705
26. Tie L,Gang Kou,Yi Peng Improving malicious URLs detection via feature engineering: Linearand nonlinear space transformation methods. Elsevier Ltd.2020, https://doi.org/10.1016/j.is.2020.10149
27. Sushant Sinha, Michael Bailey, and Farnam Jahanian. 2008. Shades of Grey: On the effectiveness of reputation-based “blacklists". In MALWARE 2008. IEEE
28. Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. 2009. Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining.
29. Sahoo, D., Liu, C., & Hoi, S. C. (2017). Malicious URL Detection using Machine Learning: A Survey. arXiv. https://doi.org/10.48550/arXiv.1701.07179
30. Ed Burns, Tech Accelerator; In-depth guide to machine learning in the enterprise: Machine Learning (30 March, 2021).
31. D. Sahoo, C. Liu, S.C.H. Hoi, “Malicious URL Detection using Machine Learning: A Survey”. CoRR, abs/1701.07179, 2017.
32. Chauhan, N. S. (2022, February 09). All you need to know about decision trees and how to build and optimize decision tree classifier.
33. Sruthi E R, Understanding Random Forest; Analytics Vidhya (27 June, 2021)
34. Christopher, A. (2021, February 02). *K-Nearest Neighbor*. Retrieved from The Startup.
35. HAYES, A. (2022, March 01). Bayes' Theorem: What It Is, the Formula, and Examples. Retrieved from Investopedia
36. Tripathy, Abinash. Sentiment Analysis Using Machine Learning Techniques. Diss. 2017.
37. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., & Venkatraman, S. (2019). Robust Intelligent Malware Detection Using.
38. Y. Chen, Y. Zhou, Q. Dong and Q. Li, "A Malicious URL Detection Method Based on CNN," 2020 IEEE Conference on Telecommunications, Optics and Computer

Science (TOCS), Shenyang, China, 2020, pp. 23-28, doi: 10.1109/TOCS50858.2020.9339761.

39. Deep Learning. IEEE Access, 7, 46717-46738. Rathore, H., Agarwal, S., Sahay, S.K., Sewak, M. (2018). Malware Detection Using Machine Learning and Deep Learning. In: Mondal, A., Gupta, H., Srivastava, J., Reddy, P., Somayajulu, D. (eds) Big Data Analytics. BDA 2018. Lecture Notes in Computer Science(), vol 11297. Springer, Cham. https://doi.org/10.1007/978-3-030-04780-1_28
40. wisdomml. (2022, July 17). Symantec Internet Security Threat Report (ISTR) 2019. Retrieved from Wisdom ML.
41. Ripa, S. P., Islam, F., & Arifuzzaman, M. (2021, July). The emergence threat of phishing attack and the detection techniques using machine learning models. In *2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI)* (pp. 1-6). IEEE.
42. Basunia, M. R., Pervin, I. A., Al Mahmud, M., Saha, S., & Arifuzzaman, M. (2020, June). On predicting and analyzing breast cancer using data mining approach. In *2020 IEEE Region 10 Symposium (TENSYMP)* (pp. 1257-1260). IEEE.
43. Bhowmik, N. R., Arifuzzaman, M., & Mondal, M. R. H. (2022). Sentiment analysis on Bangla text using extended lexicon dictionary and deep learning algorithms. *Array*, *13*, 100123.
44. Hasan, M. R., Maliha, M., & Arifuzzaman, M. (2019, July). Sentiment analysis with NLP on Twitter data. In *2019 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2)* (pp. 1-4). IEEE.
45. Toma, T., Hassan, S., & Arifuzzaman, M. (2021, July). An Analysis of Supervised Machine Learning Algorithms for Spam Email Detection. In *2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI)* (pp. 1-5). IEEE.
46. Arifuzzaman, M., Yu, K., & Sato, T. (2014, June). Content distribution in Information Centric Network: Economic incentive analysis in game theoretic approach. In *Proceedings of the 2014 ITU kaleidoscope academic conference: Living in a converged world-Impossible without standards?* (pp. 215-220). IEEE.
47. Siddiq, M. A. A., Arifuzzaman, M., & Islam, M. S. (2022, March). Phishing Website Detection using Deep Learning. In Proceedings of the 2nd International Conference on Computing Advancements (pp. 83-88).

48. Gope, J. C., Tabassum, T., Mabrur, M. M., Yu, K., & Arifuzzaman, M. (2022, February). Sentiment Analysis of Amazon Product Reviews Using Machine Learning and Deep Learning Models. In *2022 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)* (pp. 1-6). IEEE.
49. Bhowmik, N. R., Arifuzzaman, M., Mondal, M. R. H., & Islam, M. S. (2021). Bangla text sentiment analysis using supervised machine learning with extended lexicon dictionary. *Natural Language Processing Research*, *1*(3-4), 34-45.