Browser Extension for Malicious URL Detection Based on Machine Learning Model

H. Hashan Kaushalya Wijeratne
School of Computing - Cyber Security
Asia Pacific University of Technology and Innovation (APU)
Kuala Lumpur, Malaysia
tp062849@mail.apu.edu.my

Vinesha Selvarajah
School of Computing
Asia Pacific University of Technology and Innovation (APU)
Kuala Lumpur, Malaysia
vinesha@staffemail.apu.edu.my

Yogeswaran Nathan
School of Computing
Asia Pacific University of Technology and Innovation (APU)
Kuala Lumpur, Malaysia
yogeswaran.nathan@staffemail.apu.edu.my

***Abstract*— In the aftermath of the technological revolution, individuals started to depend more and more on the internet as their primary method of communication. In today's world, people use this platform for a variety of activities, including shopping, watching movies, engaging with friends on social media, getting information, and even learning. Malicious URLs are the most common source of danger in today's digital environment, according to security experts. For example, targeting people, pushing schemes, and perpetrating fraud are only a few examples of what is prohibited under the law. Malware is mostly targeted at email, with 92 percent of the 50,000 security events observed being directed towards email (Sanders, 2021). Traditionally, this recognition has been performed mostly through the use of blacklists. Blacklists, on the other hand, are not comprehensive and are unable to discover newly formed harmful URLs. In recent years, machine learning approaches have garnered more attention as a way of enhancing the generality of malicious URL detectors, which is an important goal. The purpose of the project is to provide a machine learning model with good accuracy that can be used on both the client-side and the application-side. The service will be available as a browser extension and an API, respectively.**

Keywords*— *URL, malicious URL detection, feature extraction, machine learning algorithms, browser extension, API

I. INTRODUCTION

The internet has developed into a key mode of communication as a result of the technological revolution. People from all around the globe make use of this service in their everyday lives. Shopping, watching movies, researching, social networking, and even business are all examples of activities. There are several benefits and drawbacks of internet use.

Web addresses, also known as Uniform Resource Locators (URLs), are used by users to visit particular websites. It is controlled by the webserver and is used to locate internet resources. The web server's operator handled the resource and its URL correctly (Mozilla, 2021).

Hackers are using these malicious URLs for a variety of purposes. If the victim clicks on or accesses a malicious URL, they risk installing Trojans, ransomware, key loggers, and a backdoor into their network or system. This is the primary way hackers employ to get user or organizational data.

A recent study (Sanders, 2021) found that, malware affected the world economy $500 billion in 2015, and that Figure is anticipated to reach 6 trillion dollars by 2021. By 2025, if present trends continue, costs will exceed $10 trillion. Security concerns associated to email were responsible for 92% of the 50,000 occurrences. Second highest went to "drive-by downloads" which is browser based malware. A research by Trend Micro (2020) focused at Facebook Messenger phishing and scam. Using a short URL, visitors will be sent to one of two different websites. If the user is logged into Facebook, a fake Netflix page will be shown. In Covid-19, they detected 1,025,301 malicious URLs, an increase of 47.4% from quarter 2 to the third of 2020.

II. RESEARCH BACKGROUND

Malicious URL based cyber-attacks are the most prevalent kind of cyber threat that takes place on the internet. Hackers use this method to target victims for various purposes. To collect ransom from organizations or individuals, create a backdoor on victim machine, theft personal or organizational data, and install malware application on victim machine are the instance for their purpose. Social Engineering, spam, phishing, drive-by-download are the most common malicious URL cyber threats.

Phishing URL is the most common cyber threat which attackers used to get personal or financially sensitive information. Phishing emails are one of the methods used to target the victim. It looks like originate from a trusted site and victims are deceived into revealing important information. It is possible to hijack the official website of a respectable organization by employing redirects or URL shortening services (Barracuda, 2021) to continue URL phishing and impersonation tactics. Phishing was the most common entry point for security breaches in 2019, costing an estimated $58 million (Barracuda, 2021).

Drive-by Download attack is another main attack method which hackers used to target victims. It is used to download malware into the target machine without user acknowledgement when clicking or opening a malicious URL. Even highly regarded websites are vulnerable to these sorts of attacks (Kaspersky, 2021).

A. Malicious URL Detection

There is a variety of methods that may be used to identify malicious URLs. Blacklisting, heuristics, and machine learning approaches are the most commonly used to detect malicious URLs (Vundavalli et al., 2020). The blacklisting and heuristics approaches have a major drawback. The hackers are generating continuously new URLs to target victims and avoid detection approaches. Because of that, these approaches need to update their malicious URL list often. If a malicious URL does not exist on their database, it is not detected from the blacklisting or heuristics approach. But, many anti-virus programmes employ blacklisting, even

though it has various drawbacks, because of its simplicity and effectiveness (Catak et al., 2020).

Machine learning approach using to develop prediction model by extracting URL features to detect malicious URLs. In this approach will use variety of URLs to train the model which analyse the static and dynamic properties. To provide high accuracy detection model, it needs to extract malicious URL features. There are different kind of features that may use to identify malicious URLs characteristics. They are, lexical features, blacklist features, host-based features, and content-based features (Sahoo et al., 2017).

Machine learning may be classified into three categories: supervised, unsupervised, and semi-supervised. In contrast to unsupervised machine learning, supervised machine learning makes use of data that has already been identified and labelled, while unsupervised machine learning makes use of data that has not yet been identified and labelled. Semi-supervised machine learning is a kind of machine learning in which just a portion of the data is identified and labelled. If the URL is malicious or benign, the label will indicate this (Vemuri, 2018).

B. Related Works

Numerous studies have been conducted using machine learning algorithms for detecting malicious URLs. These approaches identify malicious URLs by using a variety of machine learning algorithms and other attributes. According to (Vemuri, 2018), the Random Forest classification was used to construct the machine learning model that accurately analysed the 4.5 million shortened URLs.

The performance of Logistic Regression, Naive Bayes, and Neural Networks were compared using a machine learning model (Vundavalli et al., 2020). As a consequence, the Naive Bayes algorithm outperforms all other algorithms with a 91 percent accuracy. The Logistic Regression and Neural Network models achieved an accuracy of 86 percent and 88 percent, respectively.

The 97.3 percent accuracy reported by (Li et al., 2019) machine learning model was utilised to detect phishing websites via the use of four algorithms in combination. There are Random Forest, Gradient Boosted Decision Trees (GDBT), eXtreme Gradient Boosting (XGBoost), and Light Gradient Boosting Machine (LigthGBM).

In (Lee & Kim, 2012), a model was suggested to identify Twitter-based malicious URL detection system "WarningBird," which is a real-time service with a high detection rate. However, this approach is incapable of identifying lengthy URLs that come at irregular intervals.

The performance of Naive Bayes and Support Vector Machine (SVM) classifiers was examined in a machine learning model (Sayamber & Dixit, 2014). According to the observation, the Nave Bayes method outperformed the SVM algorithm in terms of accuracy. However, in experiments, model training took the same amount of time for both algorithms, and naive bayes reasoning is much quicker than SVM.

The URL lexical analysis approach (Darling et al., 2015) was used to identify malicious web sites. The goal of this suggested model is to identify the top limit on the accuracy rate of a lexical approach. With a 99.1% accuracy rate, 0.4 percent false positives, and an average response time of 0.62 milliseconds, it correctly identifies malicious URLs.

III. Problem Statement

In recent years, the number of Malicious URLs cyber-attacks has increased, and it has become more difficult to identify the attacks due to the fact that hackers are using a variety of ways to target various users and apps. According to a recent research (Sanders, 2021), malware cost the global economy $500 billion in 2015 and is expected to reach $6 trillion by 2021. If current trends continue, expenses will top $10 trillion by 2025. In the Covid-19 timeframe, Trend Micro (2020) discovered that 1,025,301 malicious URL assaults were detected. Furthermore, it becomes extremely difficult for users and the applications that they rely on for communication and information exchange to identify and prevent these types of cyber-attacks from occurring.

IV. Research Questions

Within the scope of this research project, the following questions will be addressed:

1) What are the current approaches to detect Malicious URLs?

2) What are the machine learning algorithms that can be used to detect malicious URLs and how can they be employed in training?

3) How would the suggested machine learning model aid in the detection and prevention of malicious URLs?

V. Aim & Objectives

A. Aim of the project

The primary aim of conducting this research study is to develop more effective malicious URL detection model which can use in real time applications in order to protect users and businesses on cyber-attacks.

B. Objectives of the project

a) To identifying the characteristics of the malicious URLs

b) To develop machine learning model API service to detect malicious URLs

c) To develop browser extension to access the machine learning model API

VI. Scope of the Project

The purpose of this study is to develop machine learning models that can be employed in a variety of applications via the use of supervised machine learning methods. To develop the model, the algorithms Logistic Regression, Random Forest, and Decision Tree will be thoroughly explored. A more accurate machine learning model for this project will be recommended based on the results of the performance comparison. It will be feasible to detect whether a certain URL is malicious or benign using the proposed model. The internet-based datasets will be utilized to train the machine learning system. This collection contains both benign and malicious URLs connected with phishing, spam, and malware. A machine learning model is supplied, along with an API that may be used to identify fraudulent URLs by any user through HTTP. The API will be developed using Python and the Django REST framework, which is the most

extensive and extensible toolkit available for developing web APIs. To make it easier for consumers to utilize this service, will create a browser extension that will connect to the API that has been built to detect malicious URLs. The accuracy of the newly created machine learning model will be compared to that of previously published models.

VII. Significance of the Project

A powerful machine learning model that can detect harmful URLs associated with phishing, spam and malware is being developed as part of this project's approach to the problem. This method's solution is intended to be used by both client-side and application-side users, and it is meant to be as simple as possible. It is planned to provide an API for machine learning models on the application side, which developers may use to create their apps that will include a malicious URL detection service, according to the API. There will be a browser extension built that will be connected with the established API to identify malicious URLs when a user accesses the online browser. The suggested machine learning model may be used by the user via the usage of this client-side browser extension-based approach.

VIII. Literature Review

People began to utilize the internet as their primary communication medium as a result of the technological revolution. Nowadays, people use this platform for a variety of activities daily, including shopping, watching movies, connecting with friends on social media, collecting information, and even learning. People may stay informed about global events and do business online thanks to the internet. People get more benefits from this platform, however, there are also many negative aspects to this platform. Here will discuss one of the harmful consequences of internet use.

Web addresses are used by internet users to locate a particular site. URLs are another name for web addresses. Uniform Resource Locator, is the acronym for URL. It identifies a certain online resource by its unique address and is controlled by the webserver. The web server owner processed the specific resource and its associated URL correctly (Mozilla, 2021).

The biggest risk in today's digital environment is from malicious URLs. It's also known as a "viral link," or "infected link," or a "weaponized link" (Gatefy, 2021). For a variety of reasons, malicious URLs like this are being generated. For example, launching targeted assaults, pushing scams, and perpetrating fraud are just a few examples. If a user clicks on this malicious link by mistake, they run the risk of downloading malware such as Trojans, ransomware, and viruses to their computer or network.

In terms of statistics and facts, Sanders (2021) highlighted in a recent study that malware cost the global economy $500 billion in 2015, a Figure that is expected to rise. Cybercrime will cost $6 trillion by 2021. Annual costs will exceed $10 trillion by 2025 if present trends continue. The majority of malware is directed at email, accounting for 92 percent of the 50,000 security incidents recorded. Browser-based malware, such as "drive-by downloads," came in the second position. Along with email spam, COVID-19 is being used in spyware, ransomware, and harmful websites because of the pandemic scenario. This is getting more prevalent as the number of infected individuals increases by the hundreds each year. Trend Micro (2020) researchers investigated phishing and fraud methods being utilized on Facebook Messenger. The user will be sent to one of two locations through a short URL. If the user is already logged into Facebook, they will be sent to a fake Netflix page. Additionally, they discovered 1,025,301 visits on malicious URLs associated with Covid-19, a 47.4 percentage increase in harmful URL hits from quarter 2 to quarter 3 of 2020.

There are many methods for detecting malicious URLs. It may be divided into two distinct parts. There are two approaches: blacklisting and machine learning (Sahoo et al., 2017). Harmful URLs are often discovered via blacklisting methods, which maintain a running list of URLs that have been identified as malicious. Each time a new URL is visited, a database search is performed. Due to the ease with which new URLs may be generated daily, blacklisting faces the challenge of maintaining a complete list of all possibly dangerous URLs. The machine learning method examines URLs and the websites that link to them to determine what information they contain. A prediction model may be trained using training data that contains both malicious and benign URLs, enabling it to differentiate between the two kinds of URLs.

The purpose of this study is to examine harmful URL detection methods and machine learning algorithms for detecting and modelling malicious URLs. The following sections comprise the paper: The second section covered URL characteristics and features, as well as malicious URL assaults. Section III addressed the techniques and procedures for detecting malicious URLs. Section IV presented a machine learning models implementation for detecting malicious URLs, and Section V summarized the study.

IX. Uniform Resource Locator

A Uniform Resource Locator (URL) is an identifier that may be used to locate a resource on the internet in an unambiguous manner. A URL is sometimes referred to as a web address. URLs include domain names and protocols that instruct a web browser how and where to locate a specific resource. End-users may access websites by typing the URL into the browser address bar, by clicking hyperlinks on online pages and social media platforms, or by sending an email containing the URL.

A. URL Structure

The URL structure is made up of two primary components: protocol identification and domain name (Sahoo et al., 2017). The protocol that will be utilized as the primary channel of access is indicated in the URL's first part. A colon and two forward slashes follow most URL protocols, however "mail to" just has a colon after it (Catak et al., 2020). Most commonly using protocols are Hypertext Transfer Protocol (HTTP), HTTP Secure (HTTPS), File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP).

HTTP is using for web sites redirection. When using this protocol, the user’s browser connects to both a server and a

site. The server is where data will be stored once it has been downloaded. An access request is sent by the browser and an access authorization response is sent by the server. The files that make up the webpage that the user wishes to view are sent along with it.

HTTPS also works like HTTP. But HTTPS protocol using SSL data layer protection which guarantees privacy and data integrity when two applications communicating over the internet. As a result, the parties involved are verified and their data is encrypted before being sent back and forth.

FTP protocol using to transfer files between systems. It all begins with the client requesting some files, and it all ends with the server supplying the file. SMTP protocol is designed for transmitting email and it can only transfer the text (Catak et al., 2020).

In the second part of the URL, the domain name or IP address is used to identify the specific internet resource. The domain name is composed of two parts: the hostname and the Uniform Resource Identifier (URI).

Fig. 1. Uniform Resource Locator Structure (Shivangi et al., 2018)

The hostname consists of a subdomain, a primary domain, and a top-level domain. A subdomain is a segment of a principal domain name that is different from the remainder of the domain name. Subdomains are used to administer and explore the website. The primary domain may create an unlimited number of subdomains or child domains (wpbeginner, 2021). The primary domain is used to refer to the website address, whereas the top-level domain (TLD) is used to refer to the last section of the domain name. TLDs are classified primarily into generic TLDs and country-specific TLDs. It was intended to convey information about the purpose and type of a domain name, as well as the geographic area from which it originated. Several prominent top-level domains (TLDs) include .com, .net, .org, .edu, and .biz. Each TLD has its registry, which is managed under the supervision of the Internet Corporation for Assigned Names and Numbers (ICANN) (Bryant, 2021). ICANN recognizes the following TLDs:

- Generic top-level domains
- Country-Code top-level domains
- Sponsored top-level domains
- Infrastructure top-level domains

The generic top-level domains (gTLDs) are the most frequently used on the internet. It contains the ".com" extension for commercial websites and the ".edu" extension for educational websites. The country code top-level domain (ccTLD) is two characters long and indicates the particular nation. Thailand's ccTLD is ".th". The sTLDs were overseen by private corporations. Infrastructure LTD has a single category known as ".arpa". The Internet Assigned Numbers Authority controls this TLD on behalf of the Internet Engineering Task Force (Bryant, 2021).

The URI identifies a resource on the web concisely and straightforwardly. It contains the directory for the file, the file's name, and the query parameters. It aids in the identification of particular websites when the user enters the URL into the browser.

B. Malicious URLs

A malicious URL has been compromised to launch a cyber-attacks. This is the most common cyber-attack on the internet. The attackers create this kind of attack for a variety of reasons. Spam, phishing, and drive-by downloads are all examples of malicious URLs and malicious websites that are used to initiate attacks. Visitors to these sites are often unaware of the dangers and fall prey to a variety of scams involving ransom, the theft of personal information (such as credit card numbers or identities), and the installation of malware. The most common malicious URL cyber-attacks are Spam, Drive-by download, Phishing and social engineering.

URL phishing is also referred to as phishing websites. Cybercriminals create phishing sites to acquire sensitive information, such as personal or financial information. Through phishing emails that seem to come from a trustworthy source, victims are duped into giving critical information. The assault is successful after a victim clicks on a link to a malicious website. The attacker then obtains any information provided by the victim. Typically, these URLs are disguised as legitimate services such as password resets or identity verification. Apart from being disguised, the website makes it very difficult for the victim to determine that it is a forgery. When victims become aware of the phishing URLs, hackers have used social engineering methods to avoid detection and dupe users into clicking malicious URLs, which compromises their systems. By using redirects or URL shortening services (Barracuda, 2021), they may hijack a legitimate company's website for their phishing campaign while continuing to use URL phishing and impersonation techniques. The majority of security breaches begin with phishing, which cost about $58 million in 2019 (Barracuda, 2021).

During a Drive-by Download Attack, malicious software is inadvertently downloaded to the target's computer or mobile device, leaving the victim exposed to future cyber-attacks. When malware is employed in an attack, it may exploit security flaws in an application, an operating system, or a web browser. Drive-by download assaults may exploit security vulnerabilities in the target's system if the victim lacks the necessary security software or updates. A virus may infect a victim's computer without the victim doing any action, such as clicking a download button or opening a malicious email attachment. These kinds of attacks are possible on any website, including respected ones (Kaspersky, 2021).

Link spam is a kind of spam that entails the addition of links that are unrelated to the content of the original post in locations such as discussion forums, blogs, and guest books. Spammers virtually never give constructive feedback; instead, they post links to their websites. Link spam's objective is to increase the number of external links to a spammer's website. The more links a page has, the higher it

is rank and therefore the higher its position in search engine results (TechTarget Contributor, 2017).

Cybercriminals are becoming more inventive in their methods of committing crimes against the people they are trying to exploit. Users' negligence and lack of knowledge are often exploited by cybercriminals. The most frequent methods used in malicious URLs to quickly target the victim are listed below.

*1) Obfuscated URLs:*URLs that have been hidden or camouflaged to mimic a genuine website's original URL are known as obfuscated. It's done to redirect visitors to a bogus website instead of the real one. To trick visitors into disclosing login and other personal information, the spoof site is frequently an exact clone of the real one (Techopedia, 2021). Users are fooled into visiting a fake site by the URL, commonly known as a hyperlink trick or a hyper-link trick. This phishing attempt is just another example of the slew of phishing scams that exist on the Internet. The four obfuscation characteristics (Vundavalli et al., 2020) are as follows:

a) obscuring the host with an IP address
b) obscuring the host with additional space
c) obscuring the host with large hostnames
d) misspelling the domain

URL obfuscation is used in conjunction with spamming to send visitors to a malicious website using a deceptive URL. To mislead visitors into accessing their site, attackers often employ a popular misspelling method in which they misspell a domain name. Malware may infect a user's PC if the URLs are disguised. The actual URLs of some pages are hidden so that visitors cannot directly access them, although it is not just used for phishing or cross-site scripting.

Redirect with annoying popups: A user may see many redirects as a result of clicking on a malicious URL. Pop-ups may be a nuisance, and users may have to put up with them. Depending on the website, these pop-ups may appear as ads or as banners. Conditional redirections are also used by attackers. Crawlers will be routed to a safe website if they click on the link. As a result, the attacker can target just regular users while fooling investigators into believing something else (Lee & Kim, 2012).

2) Misinformation Advertisements: Aiming to entice the user to click, advertisements are shown and shared on social media platforms. Some of the information in these ads may entice users to click through to the URL they promote. As for the kind of material, it may be anything from lucrative or free offers and contests to money-making schemes and so on. Ad personalization services are often used by all apps to assist guarantee that the advertisements shown to the user are relevant. Because some apps utilize new or tiny ad servers to generate more money, these ads may potentially be sources of harmful material (Shivangi et al., 2018). Hackers may simply get into these ad servers and steal valuable information. By targeting the user's interests, hackers entice them to visit malicious URLs.

3) URL Shortening: Shortened URLs are often posted on social media platforms like Twitter and Facebook. Consider the Twitter app as an example. Twitter's 140 character restriction (Catak et al., 2020) forces hackers to utilize abbreviated URLs to get beyond the security measures put in place by the service. These URLs may deceive visitors into thinking they're going to a different site altogether.

X. MALICIOUS URL DETECTION

A. Malicious URL detection approaches

Malicious URLs may be identified using a variety of methods. Malicious URLs are often detected using a combination of blacklisting, heuristics, and machine learning (Sahoo et al., 2017),(Vundavalli et al., 2020).

1) Blacklisting and Heuristics approach: Malicious URLs are often identified using well-known and commonly utilized blacklisting methods. They often update a list of harmful URLs. These risks are difficult to identify due to the continuous generation of new URLs. This is a particularly severe issue given the ease with which attackers may generate new URLs automatically. Despite the fact that blacklisting has several disadvantages, it is nevertheless used by many anti-virus applications due to its simplicity and efficacy (Catak et al., 2020).

Heuristic approaches are a subset of Blacklist techniques, intending to create a 'blacklist of signatures' (Sahoo et al., 2017). The most common attacks are recognized, and each attack type is given a signature. Intrusion Detection Systems can scan web pages for these signatures and identify unusual activity. These techniques are more generalizable than blacklisting since they can identify risks in newly created URLs as well. However, they can easily be bypassed using obfuscation methods.

Malicious URL detection using blacklisting has significant disadvantages. This method mostly utilizes malicious URLs that are already in the database. Otherwise, it's considered safe unless the URL is already in the database. To avoid being blacklisted, the vast majority of hackers often altered their malicious URLs. This is because blacklisting databases will never be able to include all harmful URLs. Because of the large database, blacklisting is very difficult. Recently, scientists have looked to Machine Learning algorithms as a possible approach to overcome this issue and develop a new method to identify malicious URLs (Sahoo et al., 2017).

2) Machine Learning approach: Malicious URLs may be detected using this technique because of the decreased risk to the user. By examining URL characteristics and particular websites, and extracts URL feature representations to builds a prediction model. This prediction algorithm is being trained on a variety of URLs, both harmful and benign. To analyze the data static features and dynamic features that may be utilized. Systems that are prospective victims may have their activity monitored using dynamic analysis methods to search for abnormalities. Machine learning methods have delved deeply into static analysis techniques. The fundamental premise is that malicious URLs have a different distribution of these characteristics than benign ones. Static analysis is the process of analyzing a website without actually running the URL. The methods used for static analysis are those in which machine learning has had the most success (Sahoo et al., 2017).

To provide a high accuracy machine learning model needs high-quality training data, which are in turn dependent on good feature extraction. Training data is essential. Next will

discuss the feature extraction methods and machine learning algorithms for malicious URLs detection.

B. Feature Extraction

The quality of feature selection is critical for obtaining a strong prediction model, and this enhance the accuracy of training data even more. Numerous functions may be accessed through a URL. Malicious URLs may be detected using a variety of characteristics.

1) Lexical Features: To identify websites, blogs, and URLs, researchers mainly utilize lexical characteristics. A URL's Lexical feature characteristics reflect some aspect of the URL's string representation. To distinguish between these characteristics, it may use the delimiters "/", "? ", or "-" (Alghamdi et al., 2017). They also take into account the URL's length and the amount of dots. When compared to genuine URLs, phishing links exhibit a different pattern of URL length.

2) Blacklist Features: Blacklists are the easiest way to detect malicious URLs. However, owing to the difficulties of maintaining complete lists, blacklisting has a fairly significant high rate of false negatives. Instead of relying only on its blacklist existence, consider utilizing it as a feature rather than a decision-maker.

3) Host-based Features: The host-name attributes of the URL are used to get host-based features (Sahoo et al., 2017). It also provide information on the whereabouts of malicious hosts, as well as their identities, management styles, and other characteristics. Location, Domain Name, and IP Address Properties are just a few of the options available.

4) Content-based Features: All the features on a website are derived by downloading it in its entirety. These are "heavy-weight" because a lot of data has to be retrieved and there may be safety issues as a result of doing so. Aspects of the HTML document at the level of the script HTML text and JavaScript use are the primary sources of features. Hackers frequently utilize JavaScript methods to encrypt harmful code. Use of eval() and unescape() extensively, for example, may indicate that encrypted code is being executed inside the HTML (Sahoo et al., 2017).

C. Machine Learning Algorithms

Detecting malicious URLs using a prediction model is easiest than other approaches. Machine learning may be used in many ways to further simplify the process. This section will discuss different machine learning methods used to develop more accurate prediction models. These algorithms are classified into two broad categories: batch learning algorithms and online learning algorithms.

1) Batch Learning Algorithms: Algorithms for batch learning assume that all of the training data is accessible before starting the training process. Numerous machine learning methods are available for training the model. Prediction accuracy varies according to their logic. The following algorithms are used for batch learning:

a) Logistic Regression: Using a classification method like Logistic Regression, data may be assigned to discrete classes. Email spam or not spam, URL malicious or benign are instances of categorization concerns. There are two types of logistic regression: Binary and Multi-linear functions fail a class (Pant, 2019).

A Machine Learning technique known as Logistic Regression is used to solve classification issues. It is based on probability theory and uses a predictive analytic method. The logistic regression hypothesis suggests that the cost function should be restricted to a value between 0 and 1. Instead of using a linear function, Logistic Regression employs a cost function that is more complicated, known as the "Sigmoid function" or the "logistic function" (Pant, 2019).

$$0 \leq h_\theta(x) \leq 1 \quad (1)$$

The sigmoid function may be used to transform any real value to a number between 0 and 1. It is used in machine learning to translate predictions into probabilities. When running the inputs through a prediction function, it should get a probability score between 0 and 1 as a result of the classifier (Pant, 2019). As an example, there have 2 classes malicious and benign. Here it assigns malicious is 1 and benign is 0. The most straightforward way of doing this is to set a cutoff value beyond which values go into Class 1 and values below which go into Class 2. In this example threshold value is assigned as 0.5. The observation would be classified as Class 1 (malicious) if the prediction function had given a 0.7 value. Class 2 (benign) observation would have prediction returned 0.2 value.

b) Naïve Bayes Classifier: In comparison to other classification techniques, Naive Bayes may be very fast (Ray, 2017). It makes use of the Bayes theorem to forecast unknown data sets. The Naive Bayes classifier comes in useful when dealing with very large datasets. It is a Bayes Bayes' Theorem-based classification technique that is predicated on the independence of predictors (Ray, 2017). The Naive Bayesian method takes the same technique for estimating the probability of a certain class based on numerous features. Although characteristics may be interdependent or reliant on one another or on the presence of other features, the Naive Bayes classifier nevertheless estimates the probability based on considering them independently (Vemuri, 2018). This technique is often used for text classification and for addressing problems that arise when dealing with a large number of classes.

c) Support Vector Machine (SVM): When it comes to classification and regression problems, the SVM supervised machine learning algorithm is the go-to solution. Most of the time, it's used to classification issues. This method uses n-dimensional space (where n is the number of features) to represent each piece of data by plotting each one as a point with a specific position. It utilizes the concept of structural risk reduction by using a maximum margin learning method, which is an instance of the normalized loss minimization framework in action.

d) Decision Tree: The Decision Tree classification method is one of the simplest and most often used. Data from real-world situations are used to create a model in the learning phase. Predicting the reaction to supplied data is done with it in the prediction phase. Decision Trees are supervised learning algorithms, and as such, they belong to the Decision Tree algorithm family (Chauhan, 2020). The use of Decision

Trees is to build a training model for predicting the class or value of a target variable. Decision trees may be classified according to the kind of target variable they are designed to analyze. Trees with categorical and continuous variables are the two kinds that may be used. A Categorical Variable Decision Tree is a Decision Tree with a Categorical Target Variable. It is termed Continuous Variable Decision Tree when the decision tree's target variable is continuously changing throughout the course of the decision process (Chauhan, 2020).

In both categorical and numerical/continuous situations, Decision Trees may be utilized. {0/1} or {Yes/No} or {True/False} are the results of categorical decision trees; numerical decision trees, on the other hand, generate a projected value. Using decision trees, users may analyze large amounts of data by splitting it into smaller chunks. The homogenous groups in the dataset are organized into these subtrees (Vemuri, 2018). When utilizing decision trees, overfitting may be a drawback. By restricting the characteristics evaluated for each subtree, the Random Forest classifier may address this problem (Vemuri, 2018).

TABLE I. Lexical Feature Group URL Features (Xuan et al., 2020)

No	Feature group	Feature	Data type	Description
1	Lexical group	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		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

TABLE II. Host-based Feature Group URL Features (Xuan et al., 2020)

No	Feature group	Feature	Data type	Description
27	Host-based feature group	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		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

TABLE III. Word-based Feature Distribution (Patil & Patil, 2018)

Sr. No	Feature name	Distribution of word based features presence in URLs	
		Benign (%)	Malicious (%)
1	Presence of ".php" word in URL string	0.03	35.66
2	Presence of "abuse" word in URL string	0.01	5.51
3	Presence of "admin" word in URL string	0.04	6.45
4	Presence of ".bin" word in URL string	0.08	0.13
5	Presence of "personal" word in URL string	0.03	0.19
6	Presence of "update" word in URL string	0.15	2.2
7	Presence of "verification" word in URL string	0.00	0.72

There are important terms related to the Decision Tree. As shown in Fig: 2, the Root Node represents the whole population or sample and is subsequently split into two or more homogenous groupings. Splitting is dividing the node into sub-nodes. If sub-node is split into additional sub-nodes, it is referred to as Decision Node. If the node is not split furthermore it is referred to as a Terminal Node.

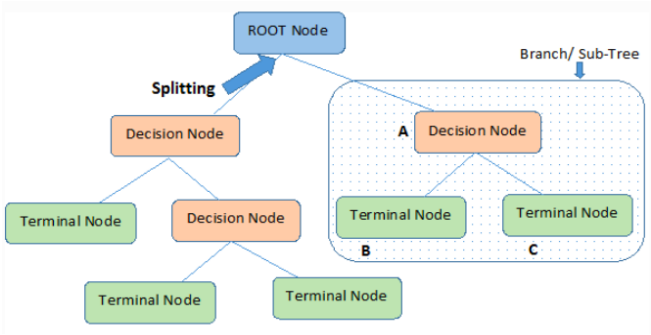

Fig. 2. Decision Tree (Chauhan, 2020)

a) Random Forest: A supervised learning method, random forest is based on decision trees. The "forest" consists of a collection of decision trees that have been trained using the bagging approach (Donges, 2021). As far as classification and regression go, it's applicable. Any training dataset with features and labels may be used in the decision tree to generate rules. The random forest method, on the other hand, uses a random selection of observations and characteristics to construct several decision trees. A mean average of the predictions made by each tree is computed at the end (Vemuri, 2018).

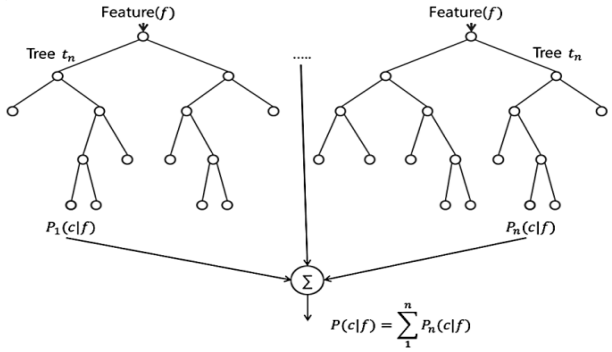

Fig. 3. Random Forest Tree (Vemuri, 2018)

Random forest hyperparameters are used to either improve the prediction power or speed of the model (Donges, 2021). A greater number of trees improves speed and predictability, but the calculation is also slowed down as a result. Random forest may be used for both classification and regression. It's also useful for seeing how much weight it gives to different input characteristics. When using the default hyperparameters, it often predicts accurately. The classification method composed of multiple decision trees is known as a "Random Forest Classifier" (Donges, 2021). Each tree in the forest is generated at random by the algorithm, which encourages the growth of forests that are unrelated to one another. The major drawback of random forest is that it may be too slow and inefficient for real-time forecasts if there are a lot of trees in the model.

2) Online Algorithms: Algorithms for online learning take data as a stream of events and build a prediction model by generating predictions and updating it to forecast the future (Catak et al., 2020).

a) First Order: This algorithm solely uses first-order features and training data to learn by updating a vector with malicious or benign labels (Catak et al., 2020).

b) Second Order: Instead of relying on first-order characteristics, this method explores second-order features such as statistical features in an attempt to improve learning efficiency (Catak et al., 2020).

Online Active Learning: Traditional methods of batch learning and online learning which are related to supervised learning often presume that learners may always get free labels for training data. Data labelling may be costly and time-consuming in actual systems, therefore this is not an option. When a requirement arises, Online Active Learning attempts to build an online learning algorithm that queries the label of an inbound unlabeled URL instance. An active learner is someone who uses online learning for a genuine system. There is a cost-sensitive online active learning (CSOAL) approach introduced for Malicious URL detection (Sahoo et al., 2017), in which an online learner decides to query a label on the fly for an incoming unlabeled URL instance so that the label will be queried just when the confidence in classifying the URL instance is low, or equivalently when making a correct prediction is highly uncertain.

XI. Machine Learning Model Implementation

There are three types of machine learning: supervised, unsupervised, and semi-supervised. Supervised machine learning uses data that has already been recognized and labelled, whereas unsupervised machine learning uses data that has not yet been identified and labelled. In semi-supervised machine learning partially data has been recognized and labelled. If URL is malicious or benign identified by the label. Supervised algorithms include Logistic Regression, Random Forest, and Decision Tree, while unsupervised algorithms, such as k-means, utilize clustering in various groups and Markov Decision Process is an example for semi-supervised machine learning algorithms (Vemuri, 2018).

A. Machine Learning Classification Metrics

The Machine Learning classification model evaluate three main metrics. They are accuracy, precision and recall (Jordan, 2017).

1) Accuracy: This key metric is defined as the proportion of accurate predictions given the test data. It may be computed simply by dividing the number of accurate predictions by the total number of predictions (JORDAN, 2017).

$$\text{accuracy} = \frac{\text{correct predictions}}{\text{total predictions}} \quad (2)$$

2) Precision: The proportion of relevant instances (true positives) in a set of all instances expected to belong to a certain class (JORDAN, 2017).

$$precision = \frac{true\ positives}{true\ positives + fasle\ positives} \quad (3)$$

3) Recall: The percentage of instances expected to belong to a class is divided by the total number of true members of the class (JORDAN, 2017).

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{fasle negatives}} \quad (4)$$

There are mainly four types of outcomes that can occur when performing the classification predictions (JORDAN, 2017). They are:

- *True Positives*: predict that an observation belongs to a class and it does
- *True Negatives*: predict that observation does not belong to a certain class and it does not belong to that class
- *False Positives*: the prediction that an observation is a member of a class when, in fact, it is not
- *False Negatives*: an observation does not belong to a class, it is predicted that it does not

TABLE IV. URLs PREDICTION CLASSIFICATION

Actual \ Prediction	**Positive**	**Negative**
Positive	TP	FN
Negative	FP	TN

B. Evaluate the Machine Learning Model

In this evaluation process, there are mainly three sections that need to focus on. They are training, testing and validating. There are various online resources available to collect datasets such as Kaggle, OpenPish, PhishTank, BlockList and from other researchers (Antonyj, 2017), (Siddhartha, 2021). To properly evaluate the model, the whole dataset is not using to train the model. To evaluate the model while it is still being built and tuning, 60 percent of data should be used for training, 20 percent for testing and, 20 percent for validation (JORDAN, 2017). There are many studies that have been conducted on the accuracy of malicious URL detection models.

According to (Vemuri, 2018), 4.5 million shortened URLs were used to build a machine learning model that used the Random Forest categorization method. It was accomplished with a 96.29 percent accuracy. The model used in (Vundavalli et al., 2020) was trained using Kaggle web source data sets. By using logical regression, Naive Bayes, and a convolutional neural network system, this study revealed which machine learning methods offer the highest detection accuracy. According to the algorithms' performance, the Logistic Regression method achieves an accuracy of 86 percent, the Neural Network algorithm achieves an accuracy of 88 percent, and the Naive Bayes algorithm achieves an accuracy of 91 percent. As a consequence, the Naive Bayes method outperformed other algorithms. (Patil & Patil, 2018) obtained a detection accuracy of 98 to 99 percent by using a decision tree machine method. It has a very low False Positive Rate and a very low False Negative Rate. The dataset was compiled using data from Alexa's Top Sites (Alexa, 2016), PhishTank's database (PhishTank, 2016), malware domain list (MalwareDomain, 2016), and jwSpamSpy (JwSpamSpy, 2016). To identify phishing websites, a stacking model was developed using a

combination of random forest and gradient boosted decision trees (GDBT), eXtreme Gradient Boosting (XGBoost), and Light Gradient Boosting Machine (LigthGBM) (Li et al., 2019). This method obtained a rate of accuracy of 97.3 percent.

XII. Project Methodology

A. Implementation of Machine Learning Model

In this approach, supervised machine learning algorithms will be used to develop the model. Random Forest, Decision Tree, and Logistic Regression are examples of supervised algorithms. The URL data utilized to construct the model has been classified as either benign or malicious.

This approach primarily focused on developing a machine learning model that will provide high accuracy for detecting malicious URLs. Additionally, to integrate this service into live systems, this model will be built as an API that users can use to develop their applications, and this service will be produced as a chrome extension that users can use as a chrome plugin. This approach can be categorized into three distinct sections as shown in Fig 4. They are as follows:

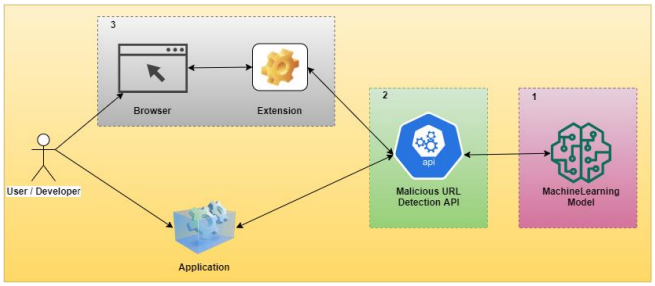

Fig. 4. Proposed Design

1) *Develop Machine Learning Model*
2) *Develop API*
3) *Develop Google Chrome Extension*

A) Develop Machine Learning Model

Machine learning algorithms such as Naïve Bayes, AdaBoost, Random Forest, and Decision Tree will be employed in this approach. Three primary metrics are evaluated by the machine learning classifying model. They are as follows: precision, accuracy, and recall (JORDAN, 2017).

Accuracy: This important measure is expressed as the ratio of correct predictions made in the face of test results. Calculating accuracy may be as simple as dividing the number of correct predictions by the total number of predicted outcomes (JORDAN, 2017).

$$\text{accuracy} = \frac{\text{correct predictions}}{\text{total predictions}} \quad (2)$$

Precision: The fraction of occurrences that are relevant (true positives) inside a collection of all instances predicted to correspond to a particular class (JORDAN, 2017).

$$precision = \frac{true\ positives}{true\ positives + fasle\ positives} \quad (3)$$

Recall: The total number of true members of a class is divided by the proportion of instances predicted to belong to the class (JORDAN, 2017).

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{fasle negatives}} \quad (4)$$

There are essentially four possible possibilities for predicting categorization results (JORDAN, 2017). They include:

- *True Positives*: forecast that an observable is a member of a class, and it is
- *True Negatives*: Forecast that observations does not match to a certain class
- *False Positives*: the inference that an observation belongs to a class when, in reality, it does not
- *False Negatives*: If an observable does not fit into a category, it is anticipated that it will

a) Data Collection

The assessment procedure will be primarily concerned with training, and testing. To test the model adequately, it requires a data collection including malicious and benign URLs. A large dataset of about 1.7M URLs were gathered from Kaggle (TIWARI, 2021), Mendeley (SINGH, 2020) (Ariyadasa et al., 2021), and UNB (UNB, 2016), all of which are freely accessible online. When a model is appropriately evaluated, the whole data set is not used for training purposes. The following data percentage will be used to assess the model. 80 percent of the total of data will be used for training, and 20 percent of the total for testing.

As seen in Fig 5, the data acquired for this study is labelled as benign, defacement, phishing, malware, malicious, and spam.

```
benign        1452815
defacement     192914
phishing       104076
malware         44086
malicious       23863
spam            12000
Name: type, dtype: int64
```

Fig. 5. Data Labelled Types

As seen in Fig 6, the dataset mentioned above is classified as benign (0) and malicious (1).

```
0    1452815
1     376939
Name: Result, dtype: int64
```

Fig. 6. Data Classification Types

However, this dataset contains just URLs, not features. As a result, it needs first to do URL feature extraction. This approach will concentrate on lexical features to extract features from URLs.

b) Feature Extraction

Due to its simplicity and accessibility, this approach focuses primarily on lexical properties. Lexical characteristics enable the model to gather and categorise the properties of a URL. All aspects are labelled in plain language and characters to aid with comprehension. As illustrated in Fig 7, this dataset will extract 20 major key features from URLs, including the domain, URL length, the presence of an IP address other than the hostname, URL shortening, the number of letters, abnormal URLs, the presence of HTTPS in URLs, and the presence of special characters such as dots, at symbol, dash, hash, per cent, and double slashes.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1829754 entries, 0 to 1829753
Data columns (total 23 columns):
 #   Column              Dtype
---  ------              -----
 0   url                 object
 1   type                object
 2   Result              int64
 3   len                 int64
 4   domain              object
 5   @                   int64
 6   ?                   int64
 7   -                   int64
 8   =                   int64
 9   .                   int64
 10  #                   int64
 11  %                   int64
 12  +                   int64
 13  $                   int64
 14  !                   int64
 15  *                   int64
 16  ,                   int64
 17  //                  int64
 18  abnormal_url        int64
 19  https               int64
 20  letters             int64
 21  Shortining_Service  int64
 22  having_ip_address   int64
dtypes: int64(20), object(3)
memory usage: 321.1+ MB
```

Fig. 7. URL Extracted Features

As seen in Fig 8, a correlational matrix for the features is used better to comprehend the relationship between target variables and predictors. This matrix is generated using the seaborn library. From the matrix's strong correlation, it may deduce the properties that may be significant in the identification of malicious URLs. Correlation free features may be omitted from the feature collection unless they just complicated the machine learning model.

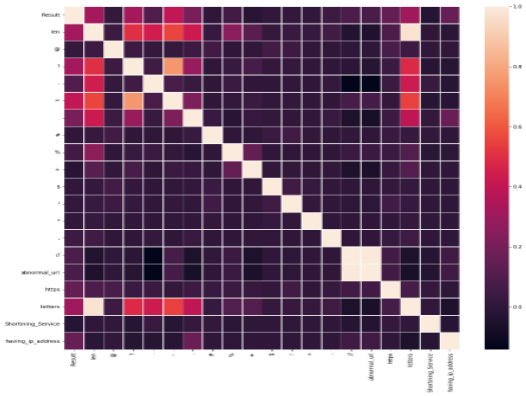

Fig. 8. Correlation Heat Map of Identified Features

Most of the algorithms cannot tolerate the null values. Therefore, before selecting the machine learning model, it needs to preprocess the data set. Preprocess is a method that deals with missing data and data formatting. Machine learning models rely heavily on this stage to get outcomes. Preprocessing may enhance the model's accuracy by minimizing the data noise.

c) Model Selection

After complete the feature extraction and preprocess, it needs to identify which machine learning algorithm is suits for this model. To determine it, the data set should train with few selected machine learning algorithms: Random Forest, Decision Tress, AdaBoost, and Naïve Bayes. The algorithm which used to train the model will based on its accuracy, training time, prediction time, fscore, precision, and recall. The algorithm will choose by comparing the results of each algorithm performance.

```
########################################
######-Model => <class 'sklearn.tree._classes.DecisionTreeClassifier'>
Test Accuracy :  91.75%
              Classification_report
              precision    recall  f1-score   support

           0       0.92      0.98      0.95    290492
           1       0.89      0.68      0.77     75459

    accuracy                           0.92    365951
   macro avg       0.91      0.83      0.86    365951
weighted avg       0.92      0.92      0.91    365951

MCC: Overall :  0.734
              Confusion_matrix
```

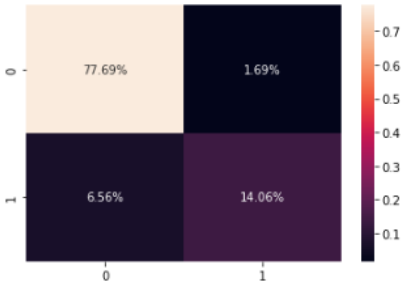

Fig. 9. Results of Decision Tree Classifier

In this approach, Decision Tree classification was compared to Random Forest to evaluate Random Forest's execution of the bagging notion. Bagging is an ensemble approach that fits numerous models to various sections of a training dataset and then combines their predictions. Random

forest is a variation on bagging in which subsets of features utilised in each data sample are also randomly selected (Brownlee, 2020). As seen in Fig 9 and Fig 10, the accuracy of Random Forest is comparatively high than the accuracy of Decision Tree.

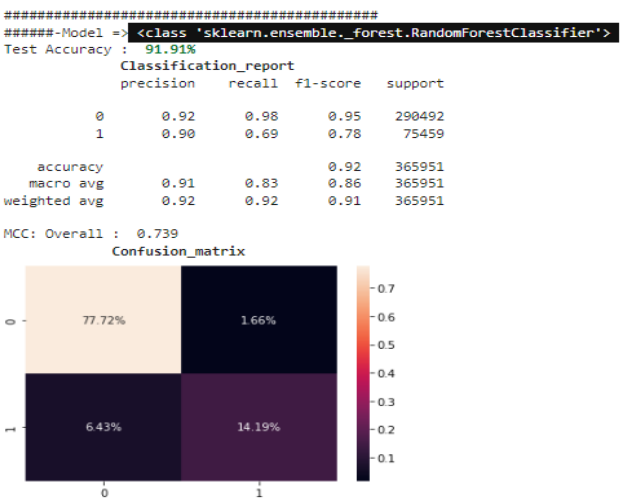

Fig. 10. Results of Random Forest Classifier

This study chose the AdaBoost classifier for analysis because it employs sequential assembly, while the Random Forest classifier used concurrent assembly (Kumar, 2019). Fig 11. illustrates the AdaBoost classification results.

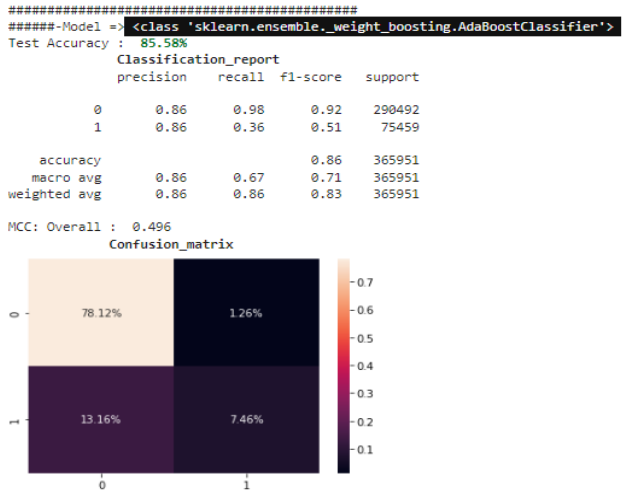

Fig. 11. Results of AdaBoost Classifier

The Naive Bayes classification method is employed in this research since it is somewhat different from more conventional models such as decision trees. Naive Bayes classification is a generative model that explains how data is produced, while a decision tree is a discriminative model that predicts the data labels (GOYAL, 2021). The results of the Naive Bayes classification are shown in Fig 12.

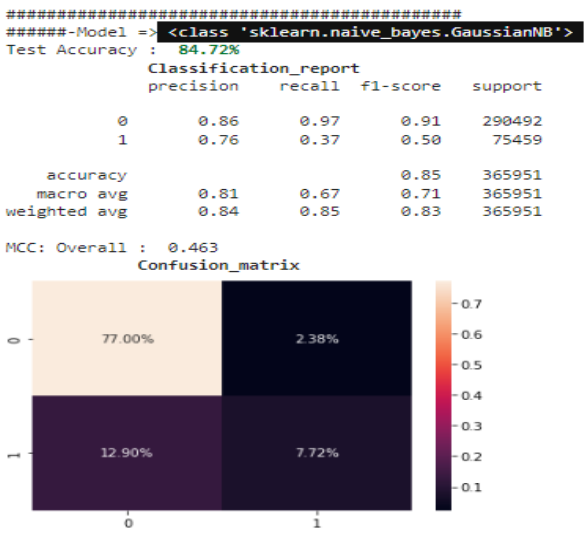

Fig. 11. Results of Naïve Bayes Classifier

The Random Forest classifier had the average best accuracy of 94.74 percent among the systems examined. To increase the accuracy of this model, several assessment approaches are used.

d) Random Forest Classifier Optimization

There are various hyperparameters in the Random Forest classifier that affect the model's behaviour on training data and prediction. Using a search process to determine a model hyperparameter configuration that performs well or ideally for a particular predictive modelling assignment is terrific. Both random and grid searches are widely used search techniques.

Specific hyperparameters should be considered while changing the Random Forest classifier. In the baseline, it used the following default hyperparameters.

- n_estimators: 100
- max_depth: None
- min_samples_split: 2
- min_samples_leaf: 1
- max_leaf_nodes: None
- min_weight_fraction_leaf: 0.0

The max_depth and n_estimators will be the emphasis of this approach. The longest route between the root node and the leaf node defines a tree's max_depth in Random Forest. This option allows setting a maximum depth restriction on how deep the random forest may develop. Number of estimators one of the most important hyperparameter in the Random Forest classifier. To improve the accuracy of previous tree predictions, the model is gradually enhanced with the addition of decision trees.

Optimize the Random Forest Classifier in this approach by using a grid Search CV with two essential hyperparameters. They are the ensemble's number of trees and the depth of each tree. The average F1-Score accuracy and the overall Matthews's Correlation Coefficient (MCC) value will be used to compare the Base-Line and best parameter classifier optimization. The MCC value is another measure that clearly depicts the score or predictions and only

gives a high score if the forecasts are accurate in all four zones of a confusion matrix. P signifies precision, and R denotes recall in the F1-score formulae.

$$F1 - Score = \frac{2PR}{(P+R)} \quad (5)$$

$$\text{MCC} = \frac{(\text{TP}*\text{TN})-(\text{FP}*\text{FN})}{\sqrt{(\text{TP}+\text{FP})(\text{TP}+\text{FN})(\text{TN}+\text{FP})(\text{TN}+\text{FN})}} \quad (6)$$

Grid Search CV returns the optimal classifier parameters, as seen in Fig 13. It returns a value of 100 for the parameter 'max depth' and a value of 300 for the parameter 'n estimators'.

```
GridSearchCV: 5 folds, 6x6 grid, timer started
RandomForestClassifier() with scoring = wtd.avg.f1_Score
        Best parameters: {'max_depth': 100, 'n_estimators': 300}
        Best CV score: 0.914
GridSearchCV Run Time 36403.27 seconds
```

Fig. 13. Best Parameter for Random Forest Classifier

B) Develop API

The machine learning model developed can be utilised to provide real-time predictions. However, the user cannot use this model directly to forecast URLs since it is difficult for the user to comprehend how it works. The strategy will design an API that will make it simple to utilise this machine learning model to address this issue. If a user submits a URL through API, it will provide a prediction indicating whether the URL is benign or malicious.

The "Django REST framework," written in Python, is used to construct this API. It is the most advanced and adaptable toolkit for creating web APIs. It features a good development community and a well-documented API. After developing the machine learning model, it is stored as a joblib extension file. This file will be added to the API project and allow for the development of the Rest API service.

To use this Rest API to predict URLs, it will need to extract URL characteristics and feed them into a mathematical machine learning model. When a user sends a URL through API, the "www." element is removed from the URL if it exists. Because specific URLs lack this tag, it is necessary first to get it to utilise the standard format for extracting URL characteristics. The API then extracts the URL's length and character count and stores them in a variable. Then it checks for special characters such as the at-symbol, hash, and dash. Extract unique character values that have been saved in variables as well. Following that, it will verify that the URL exists at an IP address other than the hostname, that the web protocol is HTPPs or HTTP, and that the URL is utilising a URL shortening service. These characteristics are also kept in Boolean format. After extracting all of the URL's data, features will send it to a machine learning model to determine if the URL is benign or malicious. When a machine learning model predicts the URL, the result is sent to the user as a response.

The REST API built in this approach is primarily aimed at developers. Developers may include this API in their apps when they want clarification of URLs that users have entered on their platform. Integrating this API enables developers to secure their applications against unauthorised activity. Developers may create their algorithms to limit risk based on the API response. The API request format is shown in Fig 14, and the API response format is depicted in Fig 15.

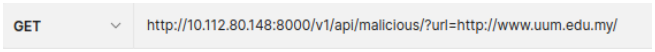

Fig. 14. GET method API request format

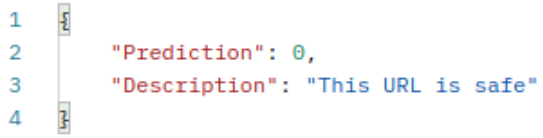

Fig. 15. API response

C) Develop Browser Extension

Nowadays, individuals are attempting to keep their knowledge current. In the past, it was done via reading books, learning from other people, and attending school. Following the technological revolution, individuals increasingly turn to digital platforms for information. Individuals attempted to acquire new information for a variety of reasons. To facilitate user interaction, the engineers created a browser application that enables users to see and interact with the content on the World Wide Web. This category includes online pages, photos, and videos. Several browser programmes exist, including Chrome, Safari, Firefox, Edge, and Opera. According to StatCounter (StatCounter, 2022), Chrome was the most used browser in 2002.

When a user types a keyword into the browser's search box, a list of particular websites with the requested information is shown. These results may be in the form of a web page, an image, or a video that particular URLs can recognise. These URLs may or may not be legitimate. Without the user's awareness, clicking on a malicious URL might have a negative effect on the user's device and personal data.

A chrome extension is developed in this approach that utilises the suggested machine learning model to identify fraudulent URLs in real-time. This chrome extension was created using JavaScript and connected with a Python REST API is used to predict URLs using a machine learning model.

The Chrome extension start with preparing a manifest.json version 3 file that basically contains all the details required for that particular, name, description, manifest version, browser actions, permissions, web accessible resources, etc. When user type an URL in the address bar or click any web URL, the extension will check the URL is legitimate or not.

As seen in Fig 16, the extension has three primary JavaScript files for executing the code. The background script is the extension's event handler; it includes listeners for critical browsing events. It remains inert until an event

occurs, at which point it executes the logic specified. Effective background scripts are loaded only when it's required and unloaded when it's become idle. When the user opens a new tab or reloads the page, it passes the current tab URL to the content script.

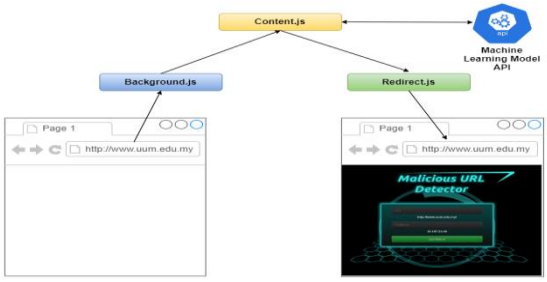

Fig. 16. Extension Architecture

The content script is used in conjunction with extensions that read or create web pages. This JavaScript file communicates the current page URL obtained from the background script file to the machine learning model API to get the predicted response. The result will be sent to the Redirect script, which contains the HTML and CSS page for the user to read. Fig 17 illustrates the redirect page that appears when a person visits a benign URL. Fig 18 illustrates the redirect page that appears when a user visits a malicious URL.

Fig. 17. Benign URL test from Extension

Fig. 18. Malicious URL test from Extension

XIII. Experimental Results

A. Result Discussion

More than 1.7 million data points were employed in this approach, together with various classifiers, for experimental purposes. This data collection contains around 1.4 million benign URLs and approximately 0.3 million malicious URLs. The whole dataset is saved in the CSV format. Each URL example includes the labels "defacement", "spam", "malware", "harmful", and "phishing" for malicious content, and "benign" for safe content. "Defacement," "malware," "spam," "malicious," and "phishing" are all labelled as 0 in feature extraction, whereas "benign" is labelled as 1.

The dataset, including safe and harmful URLs, is separated into two sections. 80% of the dataset was utilised for training, while 20% was used for testing. The experiment was done several times using classifiers from Random Forest, Decision Tree, AdaBoost, and Naive Bayes. The average performance of each classifier is shown in Table 5 after the experiment is performed five times.

TABLE V. PERFORMANCE OF DIFFERENT CLASSIFIERS

Classifier	F1-Score Accuracy	Precision	Recall	MCC
AdaBoost	0.87	0.86	0.67	0.45
Decision Tree	0.92	0.91	0.83	0.73
Naïve Bayes	0.85	0.81	0.68	0.46
Random Forest	0.94	0.93	0.86	0.78

Compared to other classifications, the findings indicate that the Random Forest Classifier produces substantial results. As a result, this approach uses Random Forest Classifier as the machine learning model for detecting malicious URLs. Fig 19 illustrates the average performance of each classification's Base-Line parameters.

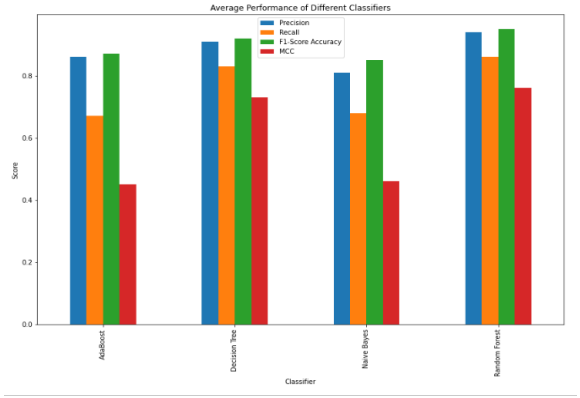

Fig. 19. Average Performance of Different Classifiers

After selecting the Random Forest classifier, the Grid Search CV method was used to identify the optimal values for hyper parameters. It returns a value of 100 for 'max depth'

and 300 for 'n estimators'. Table 6 summarises the performance of the Random Forest classifier's Base-Line parameters and the best parameters for MCC and F1-Score accuracy. According to the findings, optimised parameters outperform Base-Line parameters significantly. Fig 20 illustrate the comparison of F1-score and MCC values.

TABLE VI. RANDOM FOREST CLASSIFIER BASE-LINE AND BEST PARAMETER RESULTS COMPARISON

	Base-Line	Best-Parameter
F1-Accuracy Score	0.94	0.96
MCC	0.78	0.82

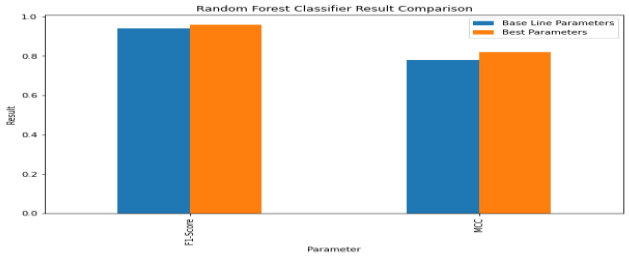

Fig. 20. Random Forest Classifier Result Comparison

XIV. CONCLUSION

The Random Forest classifier was used to develop the machine learning model for detecting malicious URLs in this approach. The empirical data in Table 2 demonstrate the performance of the machine learning classifier model that the Random Forest classifier chose. URL extracted twenty significant essential characteristics in this research to train the machine learning model. The developed machine learning model is integrated with an API and can be used to predict malicious URLs in real-time. Through the GET method, developers may leverage this API to construct their apps. The user may utilise the Chrome extension in their browser to identify dangerous URLs instantly. When the user loads a website, this chrome extension detects the browser's current tab URL and sends it as a GET request to a malicious URL detection API. According to the API answer, the user may determine whether or not the URL is dangerous. This technique is aimed at everyday consumers and developers that want protection against malicious URLs while using their applications.

REFERENCES

Alexa. (2016). *The top 500 sites on the web*. https://www.alexa.com/topsites/

Alghamdi, B., Watson, J., & Xu, Y. (2017). Toward detecting malicious links in online social networks through user behavior. *Proceedings - 2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops, WIW 2016*, 5–8. https://doi.org/10.1109/WIW.2016.41

Antonyj. (2017). *Malicious_n_Non-Malicious URL*. https://www.kaggle.com/antonyj453/urldataset/data

Ariyadasa, S., Fernando, S., & Fernando, S. (2021). *Phishing Websites Dataset*. https://data.mendeley.com/datasets/n96ncsr5g4/1

Barracuda. (2021). *URL Phishing*. https://www.barracuda.com/glossary/url-phishing

Bryant, C. (2021). *Top-Level Domain (TLD)*. https://www.techopedia.com/definition/1348/top-level-domain-tld

Catak, F. O., Sahinbas, K., & Dörtkardeş, V. (2020). *Malicious URL Detection Using Machine Learning*. 160–180. https://doi.org/10.4018/978-1-7998-5101-1.ch008

Chauhan, N. S. (2020). *Decision Tree Algorithm, Explained*. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

Donges, N. (2021). *A Complete Guide to the Random Forest Algorithm*. https://builtin.com/data-science/random-forest-algorithm

Gatefy. (2021). *What is a malicious URL?* https://gatefy.com/blog/what-malicious-url/

GOYAL, C. (2021). *Deep Understanding of Discriminative and Generative Models in Machine Learning*. https://www.analyticsvidhya.com/blog/2021/07/deep-understanding-of-discriminative-and-generative-models-in-machine-learning/#:~:text=Discriminative models draw boundaries in,the labels of the data.

JORDAN, J. (2017). *Evaluating a machine learning model*. https://www.jeremyjordan.me/evaluating-a-machine-learning-model/

JwSpamSpy. (2016). *Spam domain blacklist*. http://www.joewein.de/sw/blacklist.htm

Kaspersky. (2021). *What Is a Drive by Download*. https://www.kaspersky.com/resource-center/definitions/drive-by-download

Lee, S., & Kim, J. (2012). WarningBird: Detecting Suspicious URLs in Twitter Stream. *Network and Distributed System Security Symposium, January 2012*, 1–13. http://home.postech.ac.kr/~sangho2/%5Cnpapers3://publication/uuid/E5BD1DA0-CE72-4838-B65B-4AED755ED3B7

Li, Y., Yang, Z., Chen, X., Yuan, H., & Liu, W. (2019). A stacking model using URL and HTML features for phishing webpage detection. *Future Generation Computer Systems*, *94*, 27–39. https://doi.org/10.1016/j.future.2018.11.004

MalwareDomain. (2016). *Downloadable Lists*. http://www.malwaredomainlist.com/forums/index.php?topic=3270.0/

Mozilla. (2021). *What is a URL? - Learn web development | MDN*. https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

Pant, A. (2019). *Introduction to Logistic Regression*. https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148

Patil, D. R., & Patil, J. B. (2018). Malicious URLs detection using decision tree classifiers and majority voting technique. *Cybernetics and Information Technologies*, *18*(1), 11–29. https://doi.org/10.2478/cait-2018-0002

PhishTank. (2016). *Join the fight against phishing*. https://www.phishtank.com/

Ray, S. (2017). *6 Easy Steps to Learn Naive Bayes Algorithm with codes in Python and R*. https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

Sahoo, D., Liu, C., & Hoi, S. C. H. (2017). *Malicious URL Detection using Machine Learning: A Survey*. *1*(1), 1–37. http://arxiv.org/abs/1701.07179

Sanders, A. (2021). *15 (CRAZY) Malware and Virus Statistics, Trends & Facts*. https://www.safetydetectives.com/blog/malware-statistics/

SAVVY SECURITY. (2021). *What Is a Malicious URL?* https://cheapsslsecurity.com/blog/what-is-a-malicious-url/

Shivangi, S., Debnath, P., Saieevan, K., & Annapurna, D. (2018). Chrome Extension for Malicious URLs detection in Social Media Applications Using Artificial Neural Networks and Long Short Term Memory Networks. *2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018*, 1993–1997. https://doi.org/10.1109/ICACCI.2018.8554647

Siddhartha, M. (2021). *Malicious URLs dataset*. https://www.kaggle.com/sid321axn/malicious-urls-dataset

SINGH, A. K. (2020). *Dataset of Malicious and Benign Webpages*. https://data.mendeley.com/datasets/gdx3pkwp47/2

StatCounter. (2022). *Most Polular Browser*. https://gs.statcounter.com/

Techopedia. (2021). *Obfuscated URL*. https://www.techopedia.com/definition/4033/obfuscated-url

TechTarget Contributor. (2017). *link spam*. https://whatis.techtarget.com/definition/link-spam

Trend Micro. (2020). *Developing Story: COVID-19 Used in Malicious Campaigns*. https://www.trendmicro.com/vinfo/id/security/news/cybercrime-and-

digital-threats/coronavirus-used-in-spam-malware-file-names-and-malicious-domains \

Vemuri, P. (2018). Detecting Malicious Shortened Urls Using Machine Learning.

Vundavalli, V., Barsha, F., Masum, M., Shahriar, H., & Haddad, H. (2020). Malicious URL Detection Using Supervised Machine Learning Techniques. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3433174.3433592

Wpbeginner. (2021). *What is: Subdomain*. https://www.wpbeginner.com/glossary/subdomain/

Xuan, C. Do, Nguyen, H. D., & Nikolaevich, T. V. (2020). Malicious URL detection based on machine learning. *International Journal of Advanced Computer Science and Applications*, *11*(1), 148–153. https://doi.org/10.14569/ijacsa.2020.0110119