International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

# Detection of Phishing Websites Using Supervised Learning

**Bhanu Teja Mummadi[1], Neelakantan Puligundla*[2]**

**Submitted**: 22/07/2022 **Accepted**: 25/09/2022

***Abstract:*** Phishing attacks are becoming more common and sophisticated, putting Internet users at risk. Even though a broad range of countermeasures from academia, industry, and research have proved that these attacks are resilient, machine learning algorithms seem to be a feasible choice for distinguishing between phishing and genuine websites. Existing machine learning algorithms for phishing detection have three major drawbacks. Firstly, there is no methodology for extracting features and keeping the dataset current, nor an updated list of phishing and authentic websites. The second concern is the use of many features and the lack of evidence to justify the characteristics utilized in training the machine learning classifier. The last point of concern is the sort of datasets utilized in the research, which is skewed in terms of URL or content-based attributes. Fresh-Phish is an open-source and extensible system that extracts features and generates an up-to-date phishing dataset based on 29 distinct characteristics. The dataset includes 6,060 websites, 3,000 of which were malicious and 3,000 of which were legitimate. Therefore, 93 percent accuracy using six distinct classifiers is attained in this industry, which is a respectable maximum. To overcome the second and third difficulties, the domain name of phishing websites used to detect phishing. Based on a sample dataset, this learning model achieves 97% classification accuracy and 98% true positive rate using just 7 characteristics. This algorithm's resiliency is demonstrated. When these classifiers were tested on unidentified live phishing URLs, they detected 99.7% of them, exceeding the previous best of 95 percent.

***Keywords:*** Phishing, Machine Learning, URL Based Attributes, live Phishing URL's

## 1. Introduction

Phishing is a problem as old as the Internet itself, defined as the attempt to obtain personal information such as usernames, passwords, and credit card details, sometimes for evil purposes, by impersonating a trustworthy organization in an electronic discussion. This kind of social engineering may have disastrous effects for people's lives if they are tricked into handing up their money, credentials, or personal information. This sort of assault is often sent in the form of an email carrying the first part of the bait, hook, and catch described by Chaudhry et al.

The enticement persuades the user to click on a link. It might be a notice indicating a user's account has been hacked or otherwise deactivated, or it could be an advertising for a way to earn fast money or get illicit products. A website which looks like a legitimate company, such as a banking company or other financial institution, is often utilized as the hook. In order to trick the user into providing sensitive information, such as their account, password and credit card number, a hook is employed. The problem arises when the user submits confidential information, which the malicious website owner obtains and uses to abuse the user and his account.

Phishing assaults continue to plague individuals, internet companies, and financial institutions. In order to make money, phishing websites acquire sensitive data including usernames, passwords, pins, and even credit card numbers. Globally, phishing attempts cause an estimated $3 billion in yearly financial losses. These losses are shared by both individuals and online companies targeted by phishing attempts. Personal information that is compromised has long-term ramifications, particularly for those who utilize the Internet.

The adaptive nature of phishing attempts makes detection difficult. Making a phishing website has become easy, and attackers may easily defeat most protection techniques. Examples of severe phishing, which targets users' identities, highlight the severity and intensity of phishing assaults. Websites created using phishing toolkits may bypass practically all types of protection. Thus, it is necessary to design phishing detection methods that are strong and resilient to the phishers' adaptive techniques.

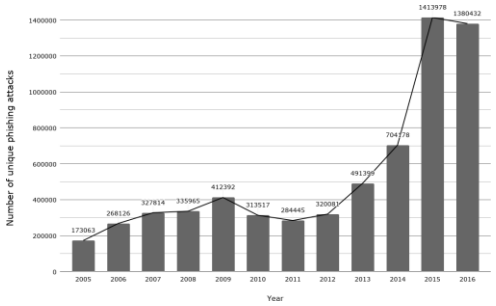


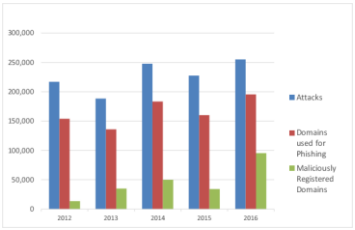

Fig 1.1: APWG documented unique assaults from 2005 to 2016

**Assaults that have been sustained throughout time:**

In Fig 1.1, each year, APWG documented the number of attacks reported. The number of assaults has constantly climbed, but in 2015, it doubled to almost 1.4 million.

According to the APWG 2016 report, there were 195,471 distinct domains used for phishing to assault targets in 2016, the highest in

any year since APWG began working. APWG also discovered 95,420 malicious domain names registered by phishers from 195,471 utilized domains. This is a record level, over three times the amount detected in 2015.

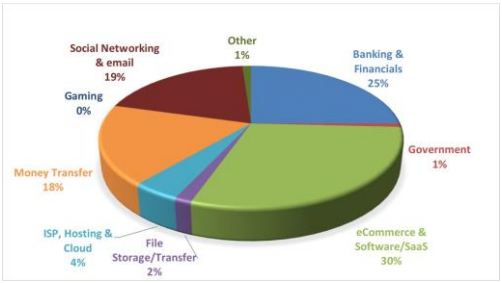



1.2 The domains used in phishing attempts from 2012 to 2016

**1.1. Adversary Aims to Inspire:**

Technology has enhanced phishing methods. Attackers now have a new reason to attack. In the last decade, phishing attacks have progressed as seen in Fig. 1.1 along with key phishing milestones. Phishing assaults targeted people, financial institutions, and military groups during the following two decades.

Attackers nowadays care less about system security and more about financial gain. On average, 30 percent of reported attacks targeted e-commerce/Software and SaaS, while 25% targeted banks and financial institutions. Identity theft accounted for 20 percent of all FTC complaints in 2009, costing victims approximately $1.7 billion.




1.3: Industry-sponsored phishing assaults in 2016

**1.2. Problem Statement:**

Based on the content of the target website, the complexity of deciphering if it is a phishing site is analysed. A phishing website usually looks and feels like a real website. The development of a machine learning classifier to distinguish between phishing and lawful websites has been accomplished.

**1.3. Limitations of Past Work:**

Modern machine learning phishing detection algorithms are email, content, and URL based. Email-based techniques analyze emails based on characteristics. However, phishing emails have evolved significantly, making these methods obsolete. Compared to other phishing approaches, spear phishing emails have a higher success rate.

Methods that analyze content and construct classifiers to identify phishing websites Search engines, DNS servers, and other third-party services are used in these works. Since there are so many training parts and so much dependence on third-party servers, this is inefficient. The user's browser history is exposed when third-party servers are used. The phishing phenomena is not effectively modelled by various parameters employed in these systems.

A variety of URL-based algorithms are used to evaluate the target URL's length, number of dots, the existence of special characters, hostname parameters such as IP address and domain age and DNS data, and geographic factors. Phishing attacks may be predicted based on URLs; however, the constant evolution of URLs means that many of the linguistic cues that these approaches look for are no longer valid. Long URLs produced by websites like Google and Amazon are an example of prolonged non-alphabetic characters in URLs. URL-based approaches will be rendered ineffective in the future due to their bias toward the datasets they use. The same issues plague hybrid detection systems that mix content and URL characteristics.

**1.4. Bias in Datasets:**

Bias in datasets occurs due to dataset utilization and URL-based characteristics. First, several researchers utilized Alexa.com to construct a tagged dataset. Anti-phishing websites, such as PhishTank.com and Openphish.com, were used instead. This is predicated on the theory that Alexa.com ranks domain names for websites, and that academics may utilize this information to create data sets from only the first pages of these websites. In contrast, anti-phishing sites include whole URLs. According to Alexa, a popular free hosting company, many phishers use www.webhost.com, a domain name that ranks well in the Alexa.com search engine. Legal websites use the URL of the home page, but phishing sites use the URL of a specific page. According to Alexa.com, most real website instances do not have subdomains, but many scam websites have.

Second, URL-based detection fails to discriminate between legal and phishing URLs. Because attackers control the URL (excluding the domain name), they may obfuscate against several methods. For example, URL properties like length, subdomains, dots ("."), unusual characters, and suspicious phrases are not always specific to phishing URLs. This explains why existing works have a high True Negative Rate (TNR).

Unbiased Intel Security datasets, no other research has particularly addressed this risk. Rather of focusing on straightforward feature design, their approach focused on reducing dataset bias. The classification accuracy of this technique is on par with that of, although requiring fewer characteristics to be considered. These methods are less likely to be spotted in the real world.

**1.5. Thesis Organization:**

The thesis continues as follows. Chapter 2 surveys known detection techniques to the challenge. Chapter 3 introduces the first approach to solving the issue, followed by trials and outcomes in Chapter 4. On examines the suggested technique and its rationale in Chapter 5.

## 2. LITERATURE SURVEY

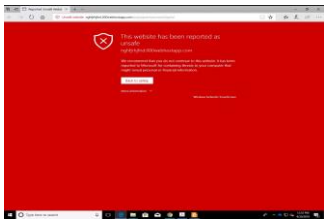
Phishing is a form of manipulating people. In this assault, the adversary hunts for unsuspecting users to draw them into the attack. An attacker, for example, creates a website that appears like a well-known email provider's login page and encourages consumers to do so. In this case, the Email provider is not a security risk. The attacker may trick the end-user if they are unaware of the possible hazards. During the previous decade, researchers attempted many ways. From a higher viewpoint, these initiatives fall into two types. In the first category, human-centered approaches to the issue are considered. The methods in this category assist end-users learn more and make better decisions when confronted with dubious websites. Second, the software-based techniques are examined. Various strategies are used to discern between legal and phishing websites, with little regard for

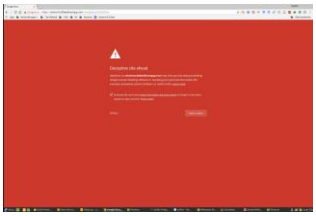
end-users. This category's output may be fed back into the first to aid end-users.

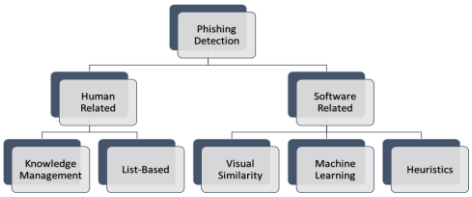
**2.1. Related to Humanity:**

Attackers employ phishing to prey on uninformed or unskilled people. Users who are unaware of the assaults are at risk.

Knowledge management lets users learn about assaults and prepare for them. But the List-based method warns the user to avoid being deceived.








2.1 Framework classification for a variety of phishing detection methods

**2.1.1 User Education and Knowledge Management:**

The people who are in danger, giving them information and helping them get better at defending themselves against these attacks is helpful.

Matthew L. Jensen and his colleagues investigated how a business could use its employees to create a "human firewall" to protect itself from phishing attacks. In order to build a phishing attack reporting and disseminating platform, they apply knowledge management research on information exchange. The findings show that knowledge management methodologies can be applied to corporate security and that phishing insights can be used to improve security. They emphasize the need of publicly acknowledging a contribution to a knowledge management system as well as having the contribution validated by the security team. They found that just performing one or the other did not increase the accuracy of phishing reports.

To minimize phishing attacks, Steve Sheng and his colleagues used learning science approaches to design and iteratively improve an online game that trains players good habits. Participants were tested on their ability to tell a phishing website from a legitimate one before and after they played the game that was produced, which reading an article and included playing the game about phishing. It turns out, based on the research, that participating in the game makes it easier for people to identify phishing sites online. Nalin Asanka et al. create a mobile version of a game that encourages home computer users to protect themselves against phishing assaults.

An extensive study was conducted to see whether workers' reactions to spear phishing emails and a range of in-person training and awareness exercises would affect the effectiveness of embedded training. Based on behavioral science data, the experiment included four distinct training conditions, each with a different message framing. There was no difference between those who had been instructed and those who had not in terms of whether they clicked on the links in a subsequent spear phishing email. The researchers were unable to determine whether the embedded training materials changed the vulnerability to spear phishing attacks since employees neglected to study the information.

**2.1.2. Based on a List:**

It is difficult to detect zero-day attacks using list-based solutions, which have a high access time but a low detection rate.

Afroz et al use fuzzy hashing methods to create profiles of credible websites. To alert users of threats, this method combines white-listing, black-listing, and a heuristic technique. Each user's whitelist of trusted websites was updated automatically by Jain and his colleagues. It warns visitors not to divulge sensitive information when accessing a site that is not on the whitelist. Scalability and dynamic updates and are problems with all list-based approaches, hence they can't be used on the client side.

These days, most browsers make use of an embedded list-based technique and periodically reload the list. When a user wants to visit a website, the browser compares it to the list, and if it is discovered, the user is alerted. Fig 2.2 is an example of the warning that Microsoft Edge and Google Chrome display to users.

Fig 2.2: Detection of a phishing assault in two separate browser sessions Microsoft Edge on the right, Google Chrome on the left.

All web pages are checked against reports of undesirable software and virus lists by Firefox before they may be visited. These lists are downloaded and updated automatically every 30 minutes by default when the "Phishing and Malware Protection" feature is turned on.

When you use Windows 10, Internet Explorer 11, or Microsoft Edge, SmartScreen performs reputation checks on the websites you visit and blocks those that it suspects of being phishing sites. Because of phishing attacks, SmartScreen also safeguards users from downloading malicious software. In Chrome, Android, and Gmail, Google Safe Browsing warns users when they are about to visit a potentially hazardous website or download malware or viruses.

**2.2. Software Related Approaches:**

In the battle against phishing assaults, relying entirely on the end-user does not lead to victory. Phishing scams tend to confuse consumers, even after they've been educated about the danger. Phishing attacks and campaigns must be prevented, detected, and eliminated via the use of software-based solutions. In this section, we'll look at a variety of software-related tactics for dealing with them, such as Visual and Textual Similarity, Machine Learning, and Heuristics.

### 2.2.1. Machine Learning:

Machine learning algorithms can find intricate correlations between data pieces of comparable kind. The learning and testing phases of many algorithms. The algorithms learn from examples and are tested for correctness.

Attackers often transmit phishing URLs through email. Detecting potentially harmful emails helps users avoid being duped by phishing websites. There is a lot of research on automated phishing email detection using the email's context. The researchers employed 16 characteristics to identify phishing emails. However, they employ email messages to extract the features, whereas concentrating on the website itself.

Using lexical and host-based properties, Ma et al. developed a system that can detect bogus URLs on the internet. A combination of URL lexical traits, registration and hosting information, and geographic location is used to classify potentially hazardous websites. Tens of thousands of characteristics associated with suspicious URLs are extracted and automatically analyzed by these algorithms. The generated classifiers identify dangerous websites using just their URLs with 95%-99% accuracy. However, their method relies on third-party services to gather host information.

Miyamoto et al. assessed nine machine learning algorithms, including Bayesian Additive Regression Trees and Support Vector Machines. Using a state-of-the-art CANTINA dataset, they examined each classifier's accuracy, and AdaBoost achieved 91.34%. The solution's resistance cannot be ensured if an updated dataset isn't used, due to the attacks' adaptive nature.

Aburrous et al. used association data-mining approaches to characterize and identify the rules for categorizing phishing websites. Six alternative categorization methods and methodologies were employed to mine the phishing training datasets. Phishing was shown using a real-world case study. For example, URLs and domain names were linked to security and encryption requirements by the rules created by their categorization model. Compared to other classic classification methods, the experimental findings showed that Associative Classification may be used successfully in practical situations.

Anti-phishing approach was suggested by researchers Xiang et al. Among the 15 points they promote are the use of search engine capabilities, third-party services, and machine learning. Additionally, two filters were constructed in order to reduce FPR and speed up the process. The first uses hashing to distinguish between almost identical phishing campaigns. Sites with no login form are marked as legitimate by a login form filter. There is a major problem with this approach since it relies on biased data. Websites that have been verified by Alexa.com are only included here. PhishTank.com's phishing websites are mostly URLs that have been shortened. As a result, each set of data is individual. Users' browsing history may also be exposed by using third-party services to extract certain functions from a website. This is a wasteful method that makes use of a disproportionate number of available features.

An algorithm developed by Verma et al. in 2015 uses the frequency and similarity of URL text characters to identify URLs that are like each other. For example, they looked at phishing URL character frequencies and suspicious word presence as characteristics. Although this method relies just on URLs, it may be biased in today's environment. They'll need to keep their features up to date when new phishing attack surfaces emerge, such as the appearance of suspicious terms.

Jain et al. proposed a machine learning strategy that solely retrieves client-side information. They discovered 19 markers that separate phishing websites from real websites. Using this method, they have a 99.49 percent success rate and 99.07 percent overall success rate in identifying phishing websites. Client-side features and third-party capabilities were not used in their method, which had certain drawbacks. Their dataset generating approach is incorrect. They got phishing URLs from PhishTank.com. They largely utilized Alexa.com, which rates the world's most popular domain names. However, Alexa.com only displays domain names, not individual sites, unlike PhishTank.com. Therefore, their characteristics are dataset biased. This was not considered while extracting features. The URL's dot count is one of its distinguishing characteristics. The phishing samples, in contrast to the genuine ones, consist of whole URLs rather than just domain names. Another function scans the URL for suspicious terms; however, many reputable websites have them.

Al-Janabi et al devised a method for spotting bogus material on social media (OSNs). Multisource characteristics have been utilized to identify fraudulent URLs in social media postings. These links will take you to dangerous websites, drive-by download attacks, phishing, spam, and scams. Using Twitter's streaming API, we were able to get this information. They used their method to just one OSN network (Twitter). Their functionalities cannot be retrieved locally, nor can users be secure while surfing outside the network.

Marchal et al. have suggested a client-side detection strategy using a browser plugin utilizing special Intel Security dataset to avoid dataset bias. They created a component that can detect a phishing web page's target website. Their method requires over 200 characteristics for classification, which takes time to extract and classify. Moreover, the dataset needed to reproduce their conclusions is not accessible.

Using URLs, source code, and third-party services, Rao et al. came up with a list of possible qualities. Their approach is ineffective and has the same drawbacks as other URL-based approaches. A user's browsing history is also exposed by using this approach, which relies on untrustworthy servers.

### 2.2.2. Similarity in both visual and textual appearance:

Because more than 92 percent of customers rely on the visual component of a website to determine the legitimacy of a website, attackers strive to make phishing websites seem as authentic as possible to deceive users. As a result, researchers utilize website similarities to distinguish between authentic and phishing sites. The usage of visual similarity is used in certain cases, while textual similarity is employed in others.

**Similarity in Appearance:**

Chen et al. suggested a method for comparing two web pages visually. They used prominent online sites to evaluate their system's real-world applicability. False positive and true positive rates were 100% accurate.

They employed Earth Mover's Distance (EMD) to compare webpages. Once they had converted the online pages into low-

resolution photos, they used color and coordinate features to express the image signatures. Then they used EMD to figure out how far off the photo signatures were from each other. To determine whether a webpage is phishing or not EMD threshold vector was used. They have created a genuine system that has captured numerous actual phishing situations.
Srinivasa Rao et al. used white-list and visual similarity approaches. In order to compare targeted and suspicious websites, they used the SURF detector to extract essential point features from each.

In order to compare phishing between two websites, each of these approaches requires a target website.

**Similarity in Text:**

A Bayesian framework was created by Zhang et al. to identify phishing web pages. The program uses linguistic and visual information to compare the protected web page to suspicious web sites. Text classifiers, picture classifiers, and a classifier fusion approach are discussed. But this method is costly and frequently yields false positives.

Extreme phishing assaults, when the phishing website closely resembles the actual website, have recently increased on financial institutions. These sites usually aim to fool visual and textual similarity detection. In the past, most content-based machine learning algorithms failed due to excessive website noise. Unlike previous research, this technique classifies websites based on a minimal set of criteria rather than their content.

**2.2.3. Heuristic Approaches:**

Neil et al. created SpoofGuard, an IE plugin that identifies phishing attempts. It assigns a score to various HTML oddities detected on webpages. The user is notified if the granted score exceeds a certain threshold, and the website is flagged as phishing. This client-side program can identify phishing websites based on page abnormalities.

During a 10-month investigation, Cui et al. monitored roughly 19000 websites for commonalities across assaults. Almost all these assaults were confirmed to be clones or variations of previously discovered ones in the database.

## 3. The First Try: A Fresh-Phish Framework

**3.1. Phishing Dataset Creation:**

The internet website characteristics dataset quickly goes out of date when it is used. The created a framework to solve this issue. This allows you to add/remove features from the dataset. The user may also rerun the extraction phase to acquire updated data for specified characteristics. Use Mohammad et al features first, then convert them to Python.

The top 3000 Alexa.com sites and the top 3000 phishtank.com sites were analyzed to create the dataset. Two things were determined: First, all Alexa.com websites are reputable websites. Due to the transitory nature of phishing websites (as indicated by the brief domain registration periods), we feel this premise is correct. For a website to get an Alexa ranking, it must be well-known and have remained so for some time. Second, we assumed that every website listed on Phishtank.com was a phishing site. phishing sites are reported to PhishTank.com by its members. Those that report phishing websites with complete accuracy are recognized and rewarded by the community which is a well-known storehouse for phishing websites.

**3.2. Implemented Features:**

To create the own dataset, taken support on the feature definitions provided by Mohammad et al. In addition to URL-based and DNS-based capabilities, there are also HTML and JavaScript-based options.

**3.2.1. Based on a URL:**

The URL of a website is critical to the functionality of any URL-based features. By hiding the URL, the attackers hope to mislead their victims. Examples of URL disguise include URLs that include IP addresses, @ signs, slashes, or prefixes/suffixes. As a result of these ways, URLs may be increased in length or subdomains can be created.

**1. Having IP Address:** Consumers may be sure that their personal information is being stolen if the URL begins with "http://125.98.4.1234/fake.html" rather than a domain name. An IP address or a valid URL were checked in Python scripts to see whether they were the same.
**2. URL length:** To assure the study's accuracy, we determined the average URL length from the data set. The findings indicated that phishing URLs have a length of 54 characters or more. Approximately 49.8% of the URLs in the sample had a length of 54 or greater.
**3. Service for reducing the length of a sentence:** It is a method used on the internet to abbreviate a URL while maintaining the same destination location. For this, a "HTTP Redirect" is used on a tiny domain name to redirect users to an enormous URL. TinyURL, for instance, is a URL shortener. When you use this service, a URL like "https://portal.had.acc.uk/" gets transformed into "bit.ly/19DX8Sk4". Other than that, it's a legitimate website.
**4. At (@) Symbol in the URL:** Since the browser normally ignores everything before the "@," a URL containing a "@" sign should not be trusted. If a URL contains the "@" sign, it is considered a phishing URL.
**5. Double Slash Redirecting:** URLs that begin with "//" are considered phishing since they direct users to a different website by use of the double slash. Phishing URLs use this tactic to hide their genuine URLs. Examples of phishing sites include "https://www.colostate.edu" and https://www.phishing.com
**6. Prefix Suffix:** It is rare to see the dash sign in URLs that are authentic. Prefixes and suffixes may be added to domain names, separated by (-), to create the appearance that the website is legitimate. Look at the website http://www.Confirme-paypal.com/. This framework checks to see whether a "-" is used in the URL name of the website. If it's being used, assuming it as a phishing website.
**7. Having Subdomain:** Assume that having the following URL: http://www.hud.ac.uk/students/. Domain names may contain country code top-level domains (ccTLDs), such as "uk" in the example. It's called a second-level domain (SLD) because of the "ac" prefix, which stands for "academic," as well as the "hud" prefix, which stands for the domain name. To extract this feature, first remove the (www.) from the URL, which is itself a subdomain. The (ccTLD) must be deleted if it exists. Finally, all the dots are connected. It is considered "Suspicious" if there are more than one dots in the URL's address. There are numerous

subdomains if there are more than two dots, and this is termed "Phishing." "Legitimate" features are those that do not have any subdomains. The number of dots in a URL was calculated. If it is larger than, then it is a phishing website; else, it is a lawful website.
**8. Portal that has not been used:** Unencrypted communication takes place over port 80, whereas encrypted communication takes place over port 443. Phishing sites are those that make use of other ports.

**3.2.2. Based on Domain Name System:**

By using information from the domain's DNS, you may get things like registration dates and length of usage.

**1. The date of the domain's most recent update:** The "Update Field" gets its data from WHOIS using this functionality. In the WHOIS database, this feature reveals when the domain owner last updated the DNS record. The real websites' WHOIS database information was updated more often than that of the fraudulent ones. If the last update was less than six months ago, consider the site to be authentic.

**2. Key for HTTPS:** A common tactic used by phishing URLs is to make them seem to be HTTPS. An example of an HTTPS-enabled URL will be http://https-colostate.edu. There has been a report that this URL is a phishing site.

**3. Time Spent in the Domain:** The WHOIS database may provide this information. Short-term presence is typical for phishing websites. After analyzing the data, the legitimate domain is at least six months old is found. The criteria for determining if a domain is legitimate or phishing is whether it has been in use for more than six months.

**4. DNS (Domain Name System) Record:** The WHOIS database may be used to get this data. It is possible that the WHOIS database does not recognize the claimed identity or that the host-name record has not been created on phishing sites. Phishing sites. It is termed phishing if the DNS record is empty or not detected; otherwise, it is evaluated as legitimate. To get DNS information from www.whoisxmlapi.com using a Python script to see whether the DNS record is empty or not.

**3.2.3. Information Obtained from Outside Sources:**

Information from outside sources rely on information gleaned from sources such as If the site is listed in Google's search index and has an Alexa page rank of at least one.

**1. The Page Rank of a web page:** Alexa rankings are considered in this section of the site. Labelling a website as phishing if it is not rated or has no traffic.
**2. Google's Web Index:** This feature determines whether a website has been included in Google's index. A website will show in search results if Google has indexed it. Because many phishing websites are only live for a short period of time, they may not be indexed by Google. Each site's Google index is identified by making a request to Google and then searching for it in the results. If Google has indexed a website, then it is legitimate in the belief. Otherwise, labelling it as phishing.

**3.2.4. Based on HTML:**

When looking at a website's HTML, there are certain important aspects that may be used to determine whether it is phishing. Favicons and images with the same source URL are instances of these characteristics. Additionally, the utilization of iFrames and the number of links pointing away from the serving domain are also HTML-based criteria.

**1. Favicon (short for "favorite icon"):** An icon (favicon) that represents a certain website is called a favicon. Favicons are often shown in the address bar of web browsers and newsreaders to help users remember which page they are on. Phishing sites use a different domain name from the one in the URL to disguise their true origins as legitimate websites. For this property, by analyzing each website's HTML code to see where the Favicon loads from. If it comes from a different domain, consider it as a phishing site.
**2. Request URL:** This function determines whether a web page's external media, such as images, videos, and music, was loaded from a separate domain. The website URL and most of the items included inside the webpage are retrieved from the same domain in authorized webpages. Using Python, a program was written which sorts all ads into two categories: domain-inside and domain-outside. For phishing purposes, if more than half of its IP addresses are from outside the domain then the site will be reported.
**3. Anchor Text for a URL:** Check out the website's backlinks using this tool. More than half the time, a website is deemed phishing if the links on it go to a domain different than the one of the websites.
**4. Tags with Links:** The <SCRIPT>, <META>, and <LINK> tags, among others, are examined by this feature to determine the domain. The site is deemed phishing if more than half of these tags point to a domain other than the sites.
**5. From the Handler:** This feature examines the behavior of the page's submit button. It's considered phishing if the site's activity is "nothing," "blank," or "about: blank." A website's URL is a sign that it is legitimate.
**6. Redirect the URL:** By designating a site as phishing if its header contains an HTML 301 redirect.
**7. Putting iFrame to Work:** HTML's <IFRAME> element was used to embed another website into the current one. Transparent <IFRAME> tags are flagged by this feature as potentially harmful to the user's experience. The website is considered a phishing site if these two conditions are met.
**8. Inbound Links to the Site:** Checks the quantity of links to your chosen site from other websites. It is considered phishing if there are no connections to the target website. This feature was not implemented; hence its score was set to neutral.

**3.2.5. Based on JavaScript:**
Features based on JavaScript look for methods to fool the user via JavaScript. Pop-up windows and ways that hide URLs or block right-clicks using JavaScript are just a few examples.

**1. Submitting to Email:** This functionality searches the submit form for a "mailto:" action. The site will be marked as phishing if it is found.
**2. When Hovering the Mouse Pointer Over:** In this method, the status bar is checked for any on-mouse-over link re-writing. Browsers are more likely to ignore this kind of scam, hence its effectiveness has decreased. Dryscrape, a Python package, was used to run web-kit in headless mode. This gives us the ability to run and test any JavaScript included in or linked to the page. Window.status JavaScript and onMouseOver are classified as phishing if they are used together.
**3. Click the Right Mouse Button:** It looks for JavaScript code that prevents the right-click operation from being performed on a

web page. It's done this way to prevent unauthorized access to the website's HTML source code. It looks for "event.button==2" in the JavaScript. phishing is detected if this happens.

**4. An Activated Pop-Up Window:** A web-kit implementation called DryScrape may be used to scrape web pages for JavaScript and HTML. A phishing site will have any of the prompt, confirm, alert or window.open methods in its JavaScript.

### 3.3. Modifying Features Definition:

Each variable in the dataset was defined by Mohammad et al. as having a binary value of 1 (meaning it was valid) or 1 (meaning it was not valid). Many of the features, such as URL length or domain age, cannot be expressed using binary values, thus they used a threshold to convert non-binary data to binary values. There are various problems with this approach. As a starting point, while defining a threshold and changing variable to binary values, significant information that may help classifiers make better selections is lost. As a second concern, it is essential that the accuracy and efficacy of the threshold be determined and regularly fine-tuned. Rather of relying on a classifier to identify "phishy" characteristics, using the actual values which were collected rather than a threshold.

### 3.4. Added Feature: Most Important Words:

You will learn more about the TF-IDF method and how it is used to develop a new feature in this section.

#### 3.4.1. TF-IDF Algorithm:

The TF-IDF is a well-known technique in information retrieval that uses a numerical statistic to measure the importance of a word in a collection or corpus. Text mining, information retrieval and user modelling all utilize it as a weighted factor.

The algorithm in this approach seeks to locate the most essential words in a document. A word's frequency is measured using the term "frequency" (TF), but the phrase "inverse document frequency" (IDF) measures the frequency of a single word over a corpus. In other words, if a phrase appears 10 times more often in a text than in the corpus, it will be given more weight than a term that appears 10 times more frequently in each. High weights are given to terms that are often used in a document but not widely used in the corpus; low weights are given to terms that are frequently used in the document but not widely used in the corpus. Each text's most important words may be found by calculating the weights of every word.

#### 3.4.2. Utilized Corpus:

In order to test the algorithms, the given text must be compared to a corpus. To conduct this study, the Open American National Corpus's Manually Annotated Sub-Corpus MASC is used. Around 500,000 words of written and spoken current American English data are included in this collection. A dataset with annotations was nevertheless used to confirm that the corpus of English phrases had a normal distribution.

#### 3.4.3. Adapting TF-IDF:

Users may examine a plain-text version of each website in the dataset, which is built in the beginning since phishers utilize plain-text versions of websites to deceive unsuspecting victims. The plain-text version lacks HTML tags and JavaScript. Next, TF-IDF algorithms are used to calculate the relative importance of each word in this version and to sort them alphabetically.

For each website, researchers look for five key phrases and compare them to the domain name. Then search using the following five keywords: We'll award the website a score of 1 if it shows in the search results. Otherwise, it will get a phishy (one point) rating. Fake websites are designed to seem authentic, with all the links pointing to a single URL. Phishers know they can't alter the domain name, so they go to great lengths to disguise it. A search for "set" will return the domain names of respectable websites, but not malicious ones. Five words as the optimal number of most crucial terms was found via through the study and statistical calculations. As a side note, Yue et al. used the same five keywords in their investigation. TF-IDF Vectorizer from Scikit-learn was used to construct this method.

### 3.5. Added Feature: Shared in Google Plus:

Many people now spend a large amount of time on social networking sites, making them an important part of daily lives. Classifiers are expected to be more accurate when data from these networks is included in the dataset. On Google Plus, how many times a site's link was spread is counted. SeoLib, a Python library, was used to get this information.

### 3.6. Content Security Policy (CSP):

The Content-Security-Policy (CSP) HTTP response header helps users avoid cross-site scripting (CSS) attacks (XSS). The administrator of the website may designate which dynamic resources can be loaded on this web page or not on the website. Web browsers must also be able to support it. According to this declaration, the browser will not load any resources that are located outside of the domain. Many websites don't support this function yet since it's a new addition.

## 4. The First Attempt: Testing and Evaluation

The framework's implementation begins by reading a CSV file containing a list of websites. URLs of websites from which feature values will be extracted and included in the final dataset are included in the input file. Websites deemed authentic by the team will be rated a1. To avoid this, put it in the number one slot. The DataLoader class must load all the URLs and labels before passing them on to the Evaluator class. The Evaluator class analyzes the URLs and produces a result vector based on their attributes. Using "URL length" as an example, the Evaluator class will calculate and save the value. An external API is used to get the values for aspects like "the Domain Age," which need external information. All features have been examined; thus, this phase will provide a vector that includes all the results. Append this comma-separated text to the dataset file at the conclusion of the framework. This procedure will be complete after the framework has gone through all the websites. One may see the formal method for this procedure in Algorithm 1.

**First Algorithm** Creating a Data Set
1: CREATE DATASET(SourceAddress)
2: URLs, DataLoader Labels
3: **while** completion of the URL list **do**
4: FeatureVector ← Evaluator.MeasureFeature(URL)
5: VectorInCSV ← CSV (FeatureVector)
6: DatasetFile ← Append (VectorInCSV)
7: **return** DatasetFile

Using Alexa and Phishtank, the information can be accessed from 6,000 domains, including 3,000 real websites and 3,000 phony ones. It was captured and included to the dataset for each characteristic.

| Machine Learning Classifier | Efficiency | AUC |
|---|---|---|
| TF Adagrad | 88.1 | 89.9 |
| TF Adadelta | 92.5 | 90.2 |
| TF GradientDescent | 92.8 | 90.2 |
| TF Linear | 82.6 | 84.7 |
| Support Vector Machine Linear | 81.3 | 83.1 |
| Support Vector Machine Guassian | 94.5 | 95.3 |

Table 4.1: Performance Evaluation of Dataset

SVM, a well-known kernel-based machine learning classifier, was used to assess the dataset's usefulness. Linear and Gaussian kernel functions were used. The trained classifiers were tested on the new data. Gradient Descent, Adagrad, and Adadelta were utilized to create a Deep Neural Network (DNN) in TensorFlow and TFcontrib utilizing built in optimization techniques. This classifier was built using TensorFlow. Cross-validation of stratified K-Fold data five times was included in this paper.

Table 4.1 shows the implementation's performance. SVM using a Gaussian kernel provides the best results. With a precision rate of 93.7%, this tool is deemed excellent.

To train the classifiers, a newly built framework is used which can extract any distinguishing attribute from a large dataset, then take that dataset and divide it into 3000 clean websites and 3000 phishing websites. All three TensorFlow-based neural networks, as well as the TensorFlow-based linear classifier and the SVM with linear and gaussian kernels, were examined. The accuracy rate was 93.7 percent.

Having this framework made it easier to extract characteristics and execute the experiment, but there are some additional problems. It is impossible to protect the privacy of consumers by using third-party services. Even more time is needed to determine which websites are legitimate and which are scams. Client-side features and strategies that do not need the participation of any third-party services are very vital for this project. TF-IDF is a dataset-dependent characteristic that is independent of data source. All these issues must be addressed with a new strategy. In the end, a functionality was created that makes use of the domain name as an input parameter.

## 5. Domain Name Based Features

A lot has been learned from Fresh-Phish in terms of phishing detection and the factors that must be considered. It is important to remember that phishing can't be understood statistically and must incorporate the attackers' desire to trick the victims. Choosing phishing detection attributes that are compatible with this concept is essential. " The content of websites is influenced by the domain name. The approach is outlined as a result of these findings. Prior to this point, there is no attention given to the domain name of the website. When you type "google.co.uk" into your browser's address bar, you'll see "Google.co.uk." appear. Search engines and DNS servers aren't used to access the website's home page.

Phishing websites may be identified by their domain names, which is why this strategy is used. A machine learning classifier is trained using sample data and many characteristics based on the domain name have been developed. A suspicious website is put to the test using the learned classifier. In the following, the recommended strategy and answers to these difficulties are detailed in detail.

First the tiny distinctions between the impact of a phishing website's domain name and URL in order to explain the design of domain name-based features. Special characters, numerical numbers, and other obfuscating elements may be used to create a URL. Phishers have a great deal of control over how the URL is formed and structured, therefore they may construct erroneous URLs that are undetectable by most machine learning algorithms. To put it another way: The adversary may construct a wide variety of URLs with one domain name, yet the domain name stays constant throughout the phishing campaign. However, even if the phisher changes the names of domain, it takes time to register name of a domain and use it for similar attacks. This is because domain name attributes are more likely to be distinct from the content of these websites. The detection algorithm will no longer heavily rely on website layout, HTML elements, or dynamic content. Phishing domain names can contain additional letters or numbers to deceive users into believing they've arrived at a legitimate website. Several harmful phishing sites, such as google.com, continue to include these quirks. These little discrepancies should be able to tell you whether a website is authentic or a phishing scam. That is why domain name-based traits are more likely to show regularity than URL-based traits.

When designing features, it's important to keep in mind that they might have an impact on the training data. In order to get around this problem, these tools make advantage of the link that exists between a domain name and the viewable content on a web page. It's not uncommon to see phishing websites with features like this one, which measures the domain's rank against all visible content on the page. Such features, however, are very difficult to implement and need extensive research on the phishing websites. Further improvements in detection accuracy are made by combining these criteria with those drawn from PhishTank.com and other community researchers' observations of phishing domain names. With this feature set, both pre-existing and brand-new functionality were included. For revamping and creating new features which are specific to the domain names used to phish for sensitive information.

The validity of the characteristics is the penultimate difficulty. In machine learning, a classifier may or may not benefit from a variety of domain-name-based attributes. Phishing and legitimate websites are both analyzed using a statistically confirmed sample of the data. Using this technique, the limit of the number of features in this classifier down to a manageable seven.

When it comes to detecting unknown or zero-day phishing attackers, the domain-based features need to be tested. A blacklist of URLs retrieved from OpenPhish.com was used to assess the classifier's performance in this area. This method was able to identify 97% to 97% of URLs in a variety of learning environments.

### 5.1. Key Contributions:

(a) Phishing detection using machine learning (ML) is based exclusively on domain name attributes. There are several benefits

to using this strategy, including the fact that it doesn't use third-party servers or search engines, suspicious terms, or URL-specific properties.
(b) The proposed approach is 96% accurate on 2000 URLs after a five-fold cross-validation.
(c) Detection rates of 97-99.7 percent on the OpenPhish.com blacklist suggest that the approach is effective against phisher-induced noise, confirming the hypothesis that datasets are skewed.
(d) Run-time detection speed is 10 times faster than the current state-of-the-art for legitimate websites using this approach.
(e) Many previous efforts in literature may need to be revisited once to show how specific characteristics like URL length might bias a learning model.

### 5.2. Domain Name Correlation for Phishing Purpose:

According to Fig 5.1, legitimate websites and phishing websites are distinct. A real picture from Amazon.com is seen in Fig 5.1; the domain name for that image is Amazonn. According to the domain name, it belongs to some impostor site with an almost identical design to Facebook.com but a different domain name. A common belief is that phishing websites strive to disguise their domain names while reputable websites aim to highlight the domain name in their webpages in order to avoid detection.

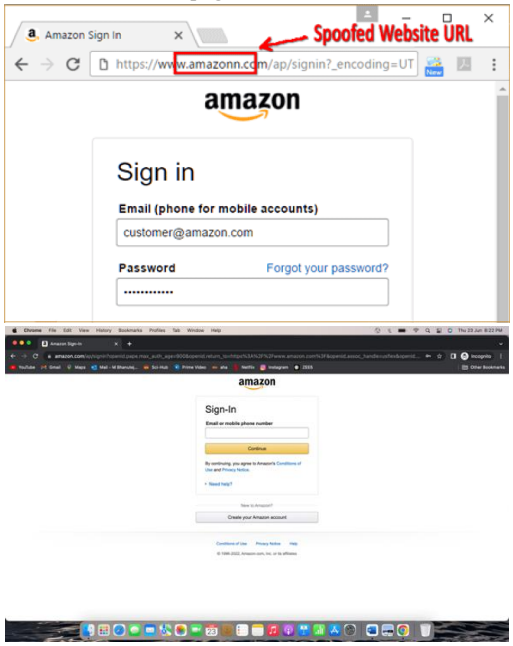



Fig:5.1 A website's domain name characteristics - Top: a biased website, bottom: a legit website

### 5.3. Engineering of Features and Testing of their validity:

In this part, the phishing efforts that make use of domain name-based features and offer statistical support for each of these attributes are detailed. The architecture of the features ensures that they are not dependent on any data values. This is done to model and decrease their dependency upon any specific data values. The challenge at hand is one that will need consideration of other elements gleaned from actual phishing assaults. Binary and non-binary features make up the feature sets; binary features have a value of 0 or 1, while non-binary features have a real-valued value. A vital aspect to keep in mind is that the domain name of a website, as well as its relationship to its content, is critical to all the feature engineering. These characteristics help to protect the features from bias in the datasets they are drawn from, as well as from noise in the data itself.

Since there are so many known phishing sites out there, the empirical cumulative distribution function (ECDF) is used to compare how well each feature worked versus the real-world equivalent. If you want to know how likely it is for the real-valued random variable X to be less than or equal to the given value of x, you need to look at the ECDF of this random variable. The feature values of the dataset were employed for the distribution along the x-axis, whilst the chance of a feature value will take a value which is less than or equal to x and it is utilized for the distribution along the y-axis. In order to determine the authenticity of a website based on its binary properties, the number of 1s that are present on the site is tallied. Non-binary attributes are used to construct ECDF plots used for phishing websites and lawful websites respectively.

Next to each feature, parentheses are placed to indicate if it is "New," meaning it was produced by us, or "Existing," meaning it was created by another researcher.

### 5.4. Domain Based Features:

#### 5.4.1. Feature 1 (present): Domain Age:

The length of time that has elapsed since the domain was first registered and made active is equal to the number of years that make up the domain's age. It is quite probable that a phishing domain will have a domain age that is much younger than that of a respectable website. The domain's age in years was estimated using WhoisXMLAPI service's Whois information and this functionality was found to be very useful throughout the testing. This function is not utilized, since it requires a third-party server, even though it has been used by other researchers in the past. Furthermore, in a sample experiment which is not present here, and obtained a startling conclusion that it had no effect on categorization accuracy.

#### 5.4.2. Feature 2 (New): Domain Length:

The longer the domain name, the more difficult it is for phishing attackers to register a domain name. The number of characters that make up the string of a domain name is referred to as its length. As can be seen in Fig 5.2, the ECDF for this feature makes a distinction that is unmistakably evident between legitimate websites and counterfeit websites.

#### 5.4.3. Feature 3 (Existing): URL Length:

One of the most often used features in phishing detection is the URL length. This is due to the common belief that fraudulent URLs tend to be longer than legal ones. The purpose of this section is to illustrate the problem of dataset bias stated in Section 1.5; thus, to explain this characteristic. In Fig 5.2 ECDF'sfeatures are shown. Most prior research have shown that results are heavily dependent on the distribution of these characteristics in phishing and authentic datasets, and this feature seems to be a terrific feature on the surface. Classification results were created with and without URL length to show how this variable affects classification. The average categorization accuracy climbs by 2% and matches the current state of the art, thanks to this innovation. The results beat those of the top researchers when feature extraction time is considered are discovered.

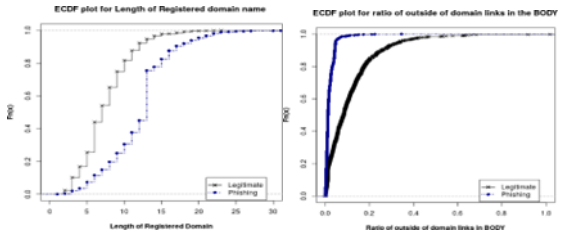


Fig 5.2: The length of a domain name

### 5.4.4. Body Link Ratio (Existing):

This attribute defines the number of links going to a single site divided by the total number of links on a page. Because of this, it is assumed that the attackers create a phishing website that mimics a legitimate one, but with another domain name. The ratio may be calculated for any phishing website that displays this behavior; thus, it doesn't matter what the site is pretending to be. Phishers may leverage well-known payment services to generate phishing pages that seem authentic except for the login form where visitors must input their personal information. When comparing a phishing site to a legitimate one, the proportion of links pointing to the current domain will change. All the links on the page are gathered and compared to see how many of those links go to the present page as opposed to all other pages on the site. Although other respectable websites demonstrated similar behavior, so a scaling approach is used to estimate the feature's worth. A value of 20 to this property if the ratio was somewhere in the [0.1, 0.2] region for a certain website is provided. Since the raw ratios are given in Fig 5.3, the ECDF of this feature has a large gap between the two distributions.

### 5.4.5. Meta-header Links are already included in Feature 5 (header)

META-description: The number of links that refer to the same domain as the total number of links on a page. Legal websites may be utilized by cybercriminals as meta tags that direct visitors away from the current domain. As can be seen in Fig 5.3, if this ratio is low, it points to a phishing website. Many reputable websites employ external connections like Google Ads, Google Analytics, and so on, making this functionality ineffective as seen in Fig 5.3. As a result, the external connections were eliminated from the consideration for inclusion in the final grouping.

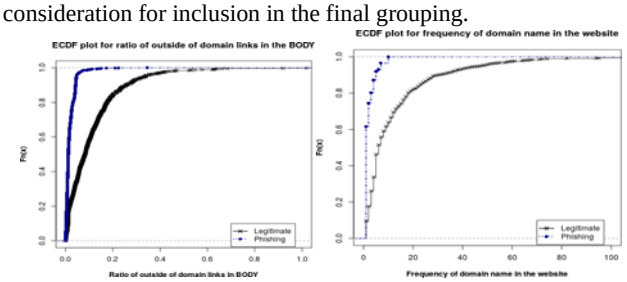


Fig 5.3: Domain Name Frequency, Link Ratio, and BODY Link Ratio

### 5.4.6. Feature 6 (New): Domain Name Frequency:

You may use this to see just how many times your domain name is mentioned on your website's content. Many internet sites, it is presumed, include the domain name in their disclaimers, privacy policies, and other sections of their website. This means that if the domain name does not appear on the webpage at all, there is a problem with the site.

Table 5.1: Distribution of binary features

| Feature | Legitimate | Phishing |
|---|---|---|
| Secure Sockets Layer (HTTPS) | 0.94 | 0.25 |
| Characters That Aren't Alphabetical | 0.07 | 0.39 |
| Matching Copyright Logos | 0.29 | 0.0 |
| Similarity in Web Page Title | 0.83 | 0.08 |

Additionally, it shows that the domain name and web page are connected. Fig 5.3 displays the ECDF for this feature. In Fig. 5.3, the ECDF is shown to have been used by other researchers with the use of search engines. Instead of relying on other servers, this definition captures a distinct essence from theirs.

### 5.4.7. There is an Existing Feature 7: Https:

A domain-specific certificate is issued. Nearly all respected websites now make use of SSL certificates and HTTPS protocol. A site with HTTPS enabled has a feature value of 1; otherwise, it has a value of zero.

### 5.4.8. Feature 8: Domain Name with Non-Alphabetical Characters

Phishing domain names are generated by the attackers using non-alphabetical characters such as digits or hyphens. It will be 1 if the domain name contains any non-alphabetic characters. Otherwise, it will be 0. There have been several studies looking at the total amount of special characters in a URL. However, the attackers may easily generate custom-based noisy URLs. The percentage distribution of binary characteristics is shown in Table 5.1.

### 5.4.9. Feature 9: Copyright Logo in Domain Name:

As a trademark ownership indicator for their company name, several respectable websites display the copyright symbol on their pages. For these kinds of websites, the domain name is often put either before or after the copyright symbol. In order to produce this feature, up to 50 characters before and after the copyright logo are considered, excluding the white spaces, and checked whether the domain name was included in the final string. As a result, this feature has a great way to set the users apart from the crowd of other web sites that employ similar techniques.

### 5.4.10. Feature 10(New): Matching of Page Title and the Domain Name:

Domain names are often duplicated in page titles, even on reputable websites. To trick viewers into thinking they were on an actual website, several phishing sites make advantage of this functionality. A phishing website does not use its own domain name as it's page headline. The suspicions were confirmed when it discovered by just 3% of phishing websites made use of this functionality, whereas over 87% of real websites did. Table 5.1 displays the sample dataset's distribution of these traits.

As a factor, the domain name is included, but they didn't depend only on it. Features 6, 10 and others relating to domain names are like all the work, and they include: the frequency with which domain names appear; the match between a domain name and its title; and others. Among other things, the technique employs more than 200 additional criteria, including domain name-based information, to make the final categorization. Domain names

matching the copyright logo in Feature 9 were totally ignored, which were found to be really useful in discovering fake sites.

## 6. Models of Machine Learning

This section provides a brief description of the classifiers and configuration parameters used in this study. Decision tree, Gaussian Naive Bayes, k-nearest-neighbors (kNN), and gradient boosting were among the classifiers tested, in addition to SVM with two alternative kernels, Gaussian and linear. To categorize an instance, the Majority Voter method is also evaluated, which relies on a majority vote. Classifiers often attempt to categorize a website as either legitimate or fraudulent.

### 6.1. SVM

Supervised learning techniques are used to divide data points into discrete categories by representing data as points in space and constructing a hyperplane in high-dimensional space. n points were classified using two distinct kernel functions: one linear, one gaussian.

$\{\overrightarrow{\mathbf{x_1}}, y_1\}, \cdots, \{\overrightarrow{\mathbf{x_n}}, y_n\}$ for each $\vec{x}$ is a d-dimensional vector and d is the number of features characterizing $\vec{x}$ and $y_i$ is for classification labels $\{-1, 1\}$ of $\overrightarrow{\mathbf{x_i}}$, The linear kernel, in general, employs a linear mathematical function, such as a linear regression, $\vec{w}.\vec{x}$ - b = 0 where b is the system parameter, to define a best hyper-plane that divides the group of points $\overrightarrow{\mathbf{x_p}}$ where y = 1 from the points where y = −1.

Similarly, Gaussian kernel which is used for radial-basis function (RBF) is defined as:

$$\mathbf{K}(\overrightarrow{\mathbf{x_i}}, \overrightarrow{\mathbf{x_j}}) = \mathbf{exp}(-\boldsymbol{\gamma}\| \overrightarrow{\mathbf{x_i}} - \overrightarrow{\mathbf{x_j}}\|^2)$$

Based on a system-defined parameter, it accomplishes the same outcome. Cost and are critical factors for the Gaussian kernel. This option defines how much of a margin there is for instances to be disregarded because they are improperly categorized. When two points are deemed "similar," the parameter modifies how similar they are and assigns them the same label. Using a grid search is a common method of determining cost and gamma parameters at the same time, which is why it's so well-known. A grid search was utilized in this study to determine the optimal cost and values.

### 6.2. Decision Tree Classifier:

Decision tree classifiers are built on tree nodes with values "learned" from training examples and branches that lead to the best possible option for the input instance, which is why they are named decision tree classifiers. There is an attribute test on another variable in a decision tree's internal node. As each internal node acts as a "splitter" to divide the instance space into smaller sub-spaces based on the attribute test, it is easy to conceptualize it in this way. There is a distinct input instance for each leaf node in the decision tree, and this correlates to a specific input instance along each branch of the tree. It has been shown that learning an ideal decision tree is an NP-hard issue. Overfitting is an issue that arises when dividing a vast attribute space using decision trees, and it is indicative of low-quality data partitioning. Small attribute spaces are ideal for decision trees. Decision trees may be quite useful for utilizing seven characteristics in this technique. Scikit-learn Python's default settings were utilized.

### 6.3. Increase in Gradient:

It's possible to construct a prediction model from a collection of imperfect predictions using the gradient boosting technique, which uses gradient descent to construct the prediction model. In order to learn the data space, the "weak" Gradient Boost Regression Tree (GBRT) model is often used. It is continuously improved by the next model, which reduces mistakes from the previous model. Gradient boosting tries to combine weak learning models into a single strong model, as shown by the example in this article:

$$F(x) = \sum_{m=1}^{M} \gamma m.\, hm(x).$$

If, $h_m$ is an iterative improvement over M trials of fixed depth GBRT, the regression parameter for that iteration will be m. The model is enhanced in the following ways at each stage:

$F_{m+1}(x) = F_m(x) + \gamma_{m+1} h_{m+1}(x)$

The $h_{m+1}$ is used to minimize the loss function L in which the current model is fitting of a data point

$X_i$: $F_m(x_i)$ as shown:

$F_{m+1}(x) = F_m(x) + argmin_h \Sigma^{n}_{i=1} L(y_i, F_m(x_i) + h(x))$

Implementation was done using Scikit Learn and the following settings were set: n_estimators = 100, Each tree has a max_depth parameter that indicates how many weak learners it can accommodate. The max_depth = 1 and learning_rate = 1.0 has set.

### 6.4. Gaussian Naïve Bayes Classifier:

For each pair of characteristics to be independent, supervised learning techniques such as Gaussian Naive Bayes classifiers are used. Briefly, given a data vector: $\vec{x} = \{x_1, x_2, \cdots, x_n\}$ having n different features, the Bayes conditional probability model assigns the following conditional probability: $P(C_k|x_1, x_2, \cdots, x_n)$, for every possible type of class $C_k$. The chain rule and the Bayesian theorem is used to construct the equation since each characteristic is independent. For the Bayes classifier, the greatest a posteriori probability is computed for each data instance, and then that class label is applied to the data instance. Naive Bayes has the advantage of using relatively minimal training data to estimate the classification parameters.

### 6.5. K-Nearest Neighbors (KNN):

There are several situations where the usage of kNN is beneficial, including security. The kNN technique determines a test case's class category by comparing it to its k-neighbors, based on the grouping of objects with comparable attributes. If the dataset is large enough, and the classification task is complex enough, k may vary.

### 6.6. Vote of Majority:

This is a special kind of classifier that is intended to play the role of a voter and decide, based on the results of other classifiers, whether a specific instance should be considered genuine. The classifier operates according to the majority-vote rule, which means that the choice made by most classifiers is the conclusion made by this classifier. To create this classifier, all the previously stated classifiers and trained them on this classifier are taken. To determine the expected value for a specific instance, testing it against all other classifiers and use the majority vote to determine the anticipated value.

## 7. Evaluation and Performance:

### 7.1. Experimentation as a Process:

After training the model using a range of machine learning classifiers, two separate tests to assess its performance were conducted. First, a predetermined dataset was employed, and then a live, unknown phishing dataset was collected from OpenPhish.com was used in the experiments. For unknown datasets, just one prior research had a detection rate of 95 percent. This method, on the other hand, has a 99.7 percent success rate in detecting suspicious activity.

### 7.2. Set of Features:

For the purpose of neutrality, the age of the domain (Feature 1) or the link ratio in the HEADER (Feature 5) didn't examine. A third-party service is required for the domain age feature, which compromises user privacy, and the HEADER feature's link ratio did not differ much between genuine and phishing datasets. Other tests were conducted with and without the URL length feature (Feature 3) included in the experiment set-up. As phishing URLs tend to be longer in public datasets, the URL length feature will show a considerable difference in distribution between phishing and authentic URLs.

True positive rate and accuracy are included in this categorization measures. This data shows how long it takes to collect the feature values from each website as well as how long it takes for training each classifier to identify phishing sites correctly. An Intel Corer 2 Duo CPU E8300Xc with 6GB of RAM and 2.4GHz clock speed is utilized in conjunction with Python 2.7 and the Sci-kit module to create this approach.

Table 7.1: The number of incidents and the source from whence they came

| # | Source | # instances | Category | Usage |
|---|---|---|---|---|
| 1 | Alexa.com | 1000 | Legitimate | Train & Test |
| 2 | PhishTank.com | 1000 | Phishing | Train & Test |
| 3 | Openphish.com | 2013 | Phishing | Test |

### 7.3. Datasets:

Three different data sources were used to collect the information and extract the features. The top 1000 most popular websites on Alexa.com are used to generate a list of reliable sites. For this research, 1,313 phishing sites from OpenPhish.com and 1,000 phishing sites from PhishTank.com were used. These datasets may be used in the next section. Even if a domain name and a web page link are both legitimate, URL length doesn't affect how these properties respond.

#### 7.3.1. DataSet 1:

1,000 trustworthy websites from Alexa.com and 1,000 dubious ones from PhishTank.com make up this bundle. The model was trained and tested using five-fold cross-validation on this dataset, which had 80% training data and 20% testing data.

#### 7.3.2. DataSet 2:

This dataset contains 3013 phishing websites, including 1000 Alexa.com domains and 3013 phishing websites are from PhishTank and OpenPhish.com. Only 1000 authentic and 1000 phishing URLs were used for training in this sample. The remaining websites from 2013 were utilized for testing purposes. Table 7.1 displays the datasets testing and training configurations.

### 7.4. Performance Metrics:

Anti-phishing detection systems using machine learning use similar metrics for evaluating the method's efficacy. TPR, TNR, PPV, ACC, and AUC were utilized to evaluate the performance of the recommended strategy. The parameters used to classify phishing and legal websites are shown in Table 7.2.

- $N_L$ is the dataset's total number of legal websites.
- $N_P$ is the dataset's total number of phishing websites.
- $N_L \rightarrow L$ stands for the total number of genuine websites.
- $N_L \rightarrow P$ stands for the total number of genuine websites classified as phishing.
- $N_P \rightarrow P$ stands for the total number of phishing websites classified as phishing.
- $N_P \rightarrow L$ stands for the total number of phishing websites classified as legitimate.

| Measure | Formula | Description |
|---|---|---|
| TPR | $\frac{N_P \rightarrow P}{N_P} \times 100$ | *correctly* classified phishing |
| TNR | $\frac{N_L \rightarrow L}{N_L} \times 100$ | *correctly* classified legitimate |
| PPV | $\frac{N_P \rightarrow P}{N_P \rightarrow P + N_L \rightarrow P} \times 100$ | *correctly* predicted phishing over *total* predicted phishing |
| ACC | $\frac{N_L \rightarrow L + N_P \rightarrow P}{N_L + N_P} \times 100$ | classified *correctly* in the dataset height |

Table 7.2: The study's performance metrics

### 7.5. DS-1 Performance in Experiment 1:

Classifiers on DS-1 were evaluated in two separate studies. The original experiment excluded just domain age, URL length, and meta-header links. It is necessary to use a third-party server in order to determine the domain's age.

In the second trial, URL length was incorporated, and the rise in classification accuracy was shown. For phishing websites, having the entire URLs, whereas for legal websites just having index pages of the domains. This bias in the dataset simply contributes to this growth.

#### 7.5.1. Results Without URL Length Feature:

With a 97 percent accuracy rate, this technique based on domain names confirms the validity of the first premise Figs 7.1 and 7.2 illustrate the outcomes of the experiments, which were run using a five-fold cross-validation. The highest value and the average value for each parameter across all validations are displayed. With an

accuracy of 99.55 percent and an average of 97 percent, one classifier performed better than the others. Gradient Boosting and Majority Voting have TPR values of 98.12 and 97.5 percent, respectively, which suggest that the classifiers can identify high-level phishing. Both classifiers have a high TNR, indicating that the learning algorithms can correctly identify authentic websites. It seems that the accuracy level of 97.74 percent is quite large in comparison to earlier research that used a wide variety of parameters.

Fig 7.1: URL length capability is not available for PPP or TNR on DS-1

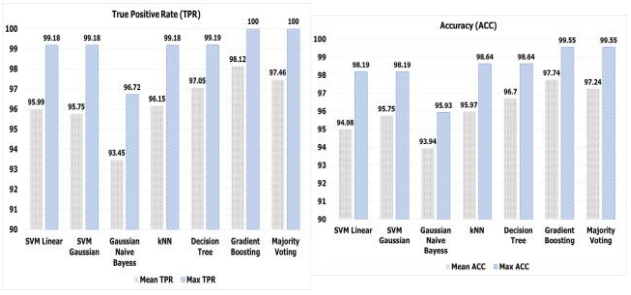



Fig 7.2: Without the URL length feature, TPR and ACC on DS-1

### 7.5.2. Results based of URL Length Feature:

The URL length feature improves the dataset's accuracy and demonstrates the data's bias.

Fig 7.3 and Fig 7.4 illustrate the outcomes of these tests. All the classifiers studied show a rising trend in all the parameters. The TNR increased from 95 percent to 98 percent across SVM linear, kNN, Decision tree, and Majority voting classifiers, which is the most important improvement. All except Gaussian Naive Bayes had a TPR of 98 percent or above, with three classifiers achieving a maximum TPR of 100 percent. Additionally, accuracy increased, with Gradient Boosting's average accuracy rising to 98.8 percent and other classifiers' maximum accuracy reaching 99.55 percent. The URL length that has a significant influence on the accuracy of the classifications, were found.

Performance metrics for each classifier may be found in Table 7.3 and Table 7.4. Each statistic's average and maximum values, except for AUC, are shown for ease of comprehension.

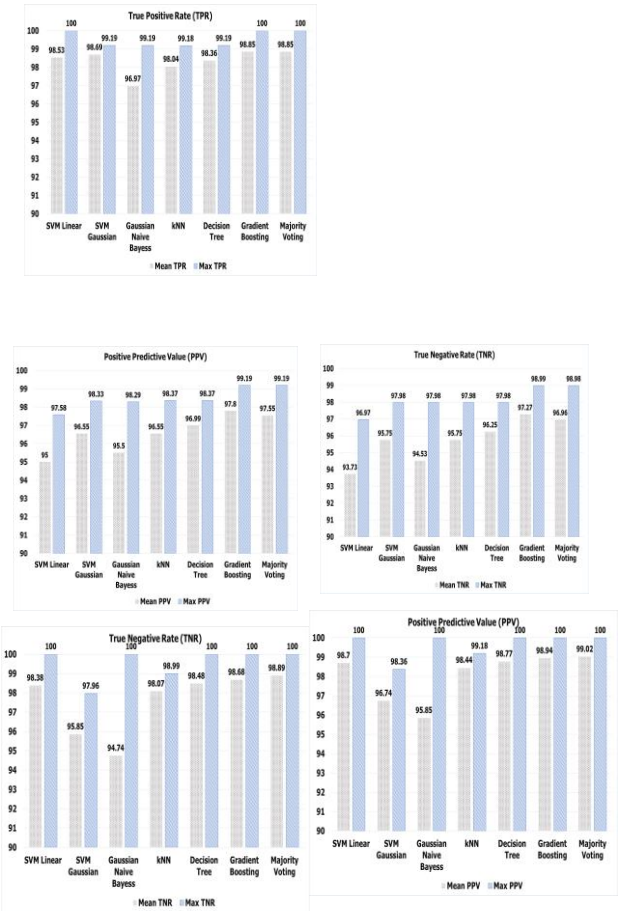



Fig 7.3: A URL length function for PPV and TNR on DS-1

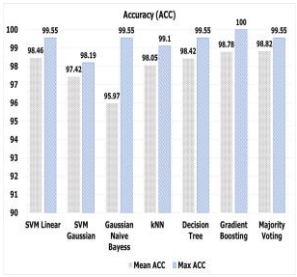



Fig 7.4: URL length feature on DS-1 for TPR and ACC

| Classifier | TPR | | TNR | | PPV | | ACC | | AUC |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Max. | Mean | Max. | Mean | Max. | Mean | Max. | Mean |
| **Support Vector Machine Linear** | 95.97 | 99.17 | 93.67 | 96.91 | 96 | 97.56 | 94.96 | 98.09 | 0.9871 |
| **Support Vector Machine Gaussian** | 95.79 | 99.16 | 95.93 | 97.93 | 96.95 | 98.32 | 95.78 | 98.29 | 0.9875 |
| **Gaussian Naive Bayes** | 92.47 | 96.75 | 94.59 | 97.96 | 95.8 | 98.26 | 93.99 | 95.83 | 0.9817 |
| **k- Nearest Neighbor** | 97.19 | 99.14 | 95.73 | 97.96 | 96.85 | 98.33 | 95.90 | 98.54 | 0.9812 |
| **Decision Tree Classifier** | 97.09 | 99.17 | 96.28 | 97.94 | 96.79 | 98.34 | 96.67 | 98.54 | 0.9878 |
| **Increase in Gradient** | 98.15 | 100 | 97.23 | 98.98 | 97.83 | 99.25 | **97.75** | 99.45 | 0.9929 |
| **Vote of the Majority** | 96.93 | 100 | 96.92 | 98.97 | 97.25 | 99.29 | **97.29** | 99.45 | 0.9913 |

Table 7.3: Measurements of DS-1 performance without considering URL length

| Classifier | TPR | | TNR | | PPV | | ACC | | AUC |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Max. | Mean | Max. | Mean | Max. | Mean | Max. | Mean |
| **Support Vector Machine Linear** | 98.56 | 100 | 98.98 | 100 | 98.6 | 100 | **98.47** | 99.75 | 0.9949 |
| **Support Vector Machine Gaussian** | 98.64 | 99.24 | 95.95 | 97.97 | 96.78 | 98.39 | **97.42** | 98.39 | 0.9914 |
| **Gaussian Naïve Bayes** | 96.96 | 99.25 | 94.84 | 100 | 95.84 | 100 | **95.94** | 99.75 | 0.9866 |
| **k-Nearest Neighbor** | 98.08 | 99.27 | 98.17 | 98.97 | 98.48 | 99.14 | **98.59** | 99.15 | 0.9908 |
| **Decision Tree Classifier** | 98.34 | 99.28 | 98.48 | 100 | 98.74 | 100 | **98.43** | 99.65 | 0.9811 |
| **Increase in Gradient** | 98.87 | 100 | 98.58 | 100 | 98.96 | 100 | **98.77** | 100 | 0.9942 |
| **Vote of the Majority** | 98.88 | 100 | 98.69 | 100 | 99.09 | 100 | **98.89** | 99.45 | 0.9946 |

Table 7.4: Metrics for DS-1 performance, including URL length

### 7.5.3. Analysis on the DS-1 performance:

A low-end desktop configuration may be used to illustrate that this approach is incredibly efficient even if the timing analysis differs dramatically from one machine to the next.

**Timing for the Feature Extraction Process:**

A few milliseconds it takes to extract a feature, which shows how effective the feature set really is.

Table 7.6 shows the results of feature extraction. It takes roughly 0.117 seconds to gather characteristics from a legitimate website, but it takes around 0.02 seconds to extract features from a malicious website, indicating the real-time nature of this method. This is a very low figure based on current state-of-the-art techniques, where extraction times are measured in seconds rather than minutes. To put this in perspective, consider how much of a difference it makes when the feature extraction takes a few milliseconds longer than a normal page load speed like msn.com.

Table 7.5: Timings for both Training and Classification

| **Features** | **Legitimate** ($\mu$s) | **Phishing** ($\mu$s) |
|---|---|---|
| HTTPS Present | 4.15 | 3.82 |
| Domain Length | 62.47 | 66.49 |
| Page Title Match | 27.3 | 32.1 |
| Frequency Domain Name | 333.6 | 33.06 |
| Non-alphabetic Characters | 32.66 | 13.62 |
| Copyright Logo Match | 2734.76 | 453.47 |
| Link Ratio in Body | 114485.87 | 19447.64 |
| URL Length | 0.3574 | 0.5062 |
| Total Time (in seconds) | 0.114 | 0.04 |

Table 7.6: Timing for Feature Extractions

**Timing for both Training and Classification:**

These classifiers can be trained and classified in only a few microseconds, demonstrating the speed and efficiency of this approach.

Extraction of features takes a significant amount of time and is not included in these testing periods. After the feature extraction, the training and testing may both be done offline and are both completed in a matter of microseconds. It is expected that this technology may be employed as a browser plug-in since feature extraction and testing takes less than 2 milliseconds.

| **Classifier** | **TPR without URL Length** | **TPR with URL Length** |
|---|---|---|
| Support Vector Machine Linear | 94.05 | 94.26 |
| Support Vector Machine Gaussian | 92.72 | 90.86 |
| Gaussian Naive Bayes | 91.08 | 92.73 |
| k-Nearest Neighbor | 93.78 | **99.5** |
| Decision Tree Classifier | 97.93 | 97.24 |
| Increase in Gradient | **98.29** | **99.73** |
| Vote of the Majority | 95.32 | 97.68 |

Table 7.7: DS-2's performance measurements, both with and without URL length restrictions

| **Classifiers** | **Training Time** (in ms) | **Testing** (in $\mu$s) |
|---|---|---|
| Support Vector Machine Linear | 1336.83 | 6.72 |
| Support Vector Machine Gaussian | 706.58 | 38.35 |
| Gaussian Naive Bayes | 2.31 | 1.42 |
| k-Nearest Neighbor | 7.33 | 14.86 |
| Decision Tree Classifier | 2.42 | 0.81 |
| Increase in Gradient | 2737.52 | 450.43 |
| Vote of the Majority | 177.76 | 3.21 |

### 7.6. DS-2 Performance in Experiment-2:

This study examines the learning method's capability to deal with previously unseen data.OpenPhish.com provided us with a list of active phishing sites for 2013. This list had more websites, but the most majority were offline or had been banned by their respective Internet Service Providers. With and without the URL length data, the classifier was trained in two different ways. The working classifier has run into its paces on a set of data from 2013. This technique has performed very well, as seen by these data. For many classifiers, TPR remained stable or even slightly improved in both tests, unlike the previous technique, which ran a similar experiment. This contrasts with the previous method, which performed a similar experiment. kNN and gradient boosting both have TPRs of 99.7% when URL length is considered. It's thus possible to identify phishing websites by their domain names.

### 7.7. Comparative work of Earlier Work:

It is shown in Table 7.8 how these results compare to existing best practices. There are several factors to consider when making comparisons: the number of features and their accuracy, whether they are client-side alone or include third-party features, and average accuracy. The run-times of the techniques were not included since they are a system-specific measure. While operating on a Core 2 Duo laptop with a low-end CPU, the system still displays microsecond-level feature extraction and categorization time.

| Approaches | #Legitimate | #Phishy | #Features | ACC | Client Node |
|---|---|---|---|---|---|
| Cantina | 2100 | 19 | 7 | 96.97 | No |
| Cantina+ | 1868 | 940 | 15 | 97 | No |
| Verma *et al.* | 13274 | 11271 | 35 | 99.3 | Partial |
| Off-the-Hook | 20000 | 2000 | 214 | 99.97 | Yes |
| App without length of URL | 1000 | 3015 | 9 | 97.5 | Yes |
| App with URL Length | 1000 | 3015 | 7 | 98.5 | Yes |

Table 7.8: A comparison with the most recent methods of research

## 8. Conclusion and Work to be done in Future:

### 8.1. Conclusion:

The issue of phishing detection is investigated using several methods that are based on machine learning in this research. This was the initial effort, and it was termed the Fresh-Phish Framework because the issue statistical. Taking an issue solely statistically is inadequate for solving it properly. In order to find a workable solution, it is also necessary to understand the motives of the phishing attacker.

Machine learning techniques may be used to identify phishing websites by simply utilizing domain name-based attributes. Using multiple phishing datasets and genuine web sites avoided any possible classification bias. Because considering the relationship between the domain name and the phishing purpose, this method is one of a kind. To get a 97 percent categorization rate, just seven characteristics are used. Detection rates for OpenPhish.com blacklisted live URLs ranged from 97 to 99.7 percent. To counter the sophisticated methods used by phishers to avoid detection, this technique has shown to be flexible. If an attacker can get around this categorization system, they will have to put in a lot of effort to do so. For the sake of evading the tactic, an adversary may create a website that raises red flags among users about its true objective. Although it seems to be a more accurate method, it may not be implemented for some time. The short time it takes to extract and classify features suggests that this approach can be used in real time. When it comes to the most advanced and sophisticated phishing schemes, this method is likely to be very successful.

### 8.2. Future Work:

Machine learning methods for phishing detection will be examined in the future for their ability to withstand newer assaults. When one visits a dubious site, an add-on browser is created that will notify.


## Acknowledgements

I Bhanu Teja pursuing MTech CSE at VNR VJIET is thankful to Professor Dr. P. Neelakantan at VNR VJIET, Department of CSE for guiding me the concepts of Machine learning with applications to phishing websites.


## Author contributions

**Bhanu Teja Mummadi:** Proposed and presented machine learning algorithms for Phishing Websites
**Neelakantan Puligundla:** Guidance of Machine Learning Concepts and Algorithms.

## Conflicts of interest

The authors declare no conflicts of interest.

## References


[1] Salloum, Said, et al. "A Systematic Literature Review on Phishing Email Detection Using Natural Language Processing Techniques." *IEEE Access* (2022).

[2] Sánchez-Paniagua, Manuel, et al. "Phishing URL Detection: A Real-Case Scenario Through Login URLs." *IEEE Access* 10 (2022): 42949-42960.

[3] Abdillah, Rahmad, et al. "Phishing Classification Techniques: A Systematic Literature Review." *IEEE Access* (2022).

[4] Garg, D. K. . (2022). Understanding the Purpose of Object Detection, Models to Detect Objects, Application Use and Benefits. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 01–04. https://doi.org/10.17762/ijfrcsce.v8i2.2066

[5] Assefa, Amanuel, and Rahul Katarya. "Intelligent Phishing Website Detection Using Deep Learning." *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*. Vol. 1. IEEE, 2022.

[6] Jaber, Aws Naser, Lothar Fritsch, and Hårek Haugerud. "Improving Phishing Detection with the Grey Wolf Optimizer." *2022 International Conference on Electronics, Information, and Communication (ICEIC)*. IEEE, 2022.

[7] BOUIJIJ, Habiba, and Amine BERQIA. "Machine Learning Algorithms Evaluation for Phishing URLs Classification." *2021 4th International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*. IEEE, 2021.

[8] Garg, D. K. . (2022). Understanding the Purpose of Object

Detection, Models to Detect Objects, Application Use and Benefits. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 01–04. https://doi.org/10.17762/ijfrcsce.v8i2.2066

[9] Rasool, Saira Banu Mohammed, M. Gnanaprakash, and M. SenthilMurugan. "A Prediction of Phishing Websites by Optimal Feature Extraction using Recurrent Neural Network." *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2022.

[10] Do, Nguyet Quang, et al. "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions." *IEEE Access* (2022).

[11] Kiran, M. S., & Yunusova, P. (2022). Tree-Seed Programming for Modelling of Turkey Electricity Energy Demand. International Journal of Intelligent Systems and Applications in Engineering, 10(1), 142–152. https://doi.org/10.18201/ijisae.2022.278

[12] Livara, Arriane, and Rowell Hernandez. "An Empirical Analysis of Machine Learning Techniques in Phishing E-mail detection." *2022 International Conference for Advancement in Technology (ICONAT)*. IEEE, 2022.

[13] Chai, Yidong, et al. "An explainable multi-modal hierarchical attention model for developing phishing threat intelligence." *IEEE Transactions on Dependable and Secure Computing* 19.2 (2021): 790-803.

[14] Garg, D. K. . (2022). Understanding the Purpose of Object Detection, Models to Detect Objects, Application Use and Benefits. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 01–04. https://doi.org/10.17762/ijfrcsce.v8i2.2066

[15] Valecha, Rohit, Pranali Mandaokar, and H. Raghav Rao. "Phishing email detection using persuasion cues." IEEE transactions on Dependable and secure computing 19.2(2021): 747-756.

[16] Sun, Xiaoqiang, F. Richard Yu, and Peng Zhang. "A survey on cyber-security of connected and autonomous vehicles (CAVs)." *IEEE Transactions on Intelligent Transportation Systems* (2021).

[17] Gopal, S. B., et al. "Autoencoder based Architecture for Mitigating Phishing URL attack in the Internet of Things (IoT) using Deep Neural Networks." *2022 6th International Conference on Devices, Circuits and Systems (ICDCS)*. IEEE, 2022.

[18] Garg, D. K. . (2022). Understanding the Purpose of Object Detection, Models to Detect Objects, Application Use and Benefits. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 01–04. https://doi.org/10.17762/ijfrcsce.v8i2.2066

[19] Venugopal, Shreya, et al. "Detection of Malicious URLs through an Ensemble of Machine Learning Techniques." *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. IEEE, 2021.

[20] Suleman, Muhammad Taseer, and Amir Ali. "Detection of Phishing Websites through Computational Intelligence." *2021 International Conference on Innovative Computing (ICIC)*. IEEE, 2021.

[21] Garg, D. K. . (2022). Understanding the Purpose of Object Detection, Models to Detect Objects, Application Use and Benefits. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 01–04. https://doi.org/10.17762/ijfrcsce.v8i2.2066