Journal of Theoretical and Applied Information Technology
31st August 2025. Vol.103. No.16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A SMART ANTI-PHISHING MODEL FOR PHISHING WEBSITE DETECTION USING MACHINE LEARNING APPROACHES BASED ON HYBRID FEATURES

M.VENKATA KRISHNA REDDY[1] , S.CHINA RAMU[1], P.RAMESH BABU[2], P.NIRUPAMA[3], B.RAMAKANTHA REDDY[4], KADIYALA RAMANA[5]

[1]Department of CSE, Chaitanya Bharathi Institute of Technology, Gandipet, Hyderabad, India
[2]Department of IT, Chaitanya Bharathi Institute of Technology, Gandipet, Hyderabad, India
[3]Department of CSE, Vemu Institute of Technology, P.Kothakota, Chittoor, Andhra Pradesh, India
[4]Department of CSE(AIML), Sri Venkateswara college of engineering, Tirupati, Andhra Pradesh,India
[5]Department of AI&DS, Chaitanya Bharathi Institute of Technology, Gandipet, Hyderabad, India
E-mail: [1]krishnareddy_cse@cbit.ac.in, chinaramu@gmail.com, [2]palamakularamesh@gmail.com, [3]nirupamacse@vemu.org, [4]ramakanthareddy@gmail.com, [5]ramana.it01@gmail.com

ABSTRACT

Phishing is a major concern in a changing society. The rise of the Internet has led to a new form of data theft, known as cybercrime. One of the most prevalent forms of cybercrime, phishing attempts to trick users into divulging personal information by creating a convincing look and feel of a trusted online service, like a bank, grocery store, or online media website. The problem of detecting phishing websites has been discussed on multiple platforms, with approaches varying from straightforward classifiers to complex hybrid systems. A novel phishing detection system, "Phishing URL Detection, PUD", is proposed here. It uses machine learning approaches to analyze results from various methods applied to URLs and validates them against existing research. URL-based phishing is a prevalent method to collect user data when accessing a malicious website. Detecting rogue URLs is difficult. The proposed method seeks to discover such websites using machine-learning approaches that analyze the behavior and attributes of the suggested URL. To better understand the structure of malicious URLs, various machine-learning methods were tested for feature evaluation. Precise parameter tuning facilitates choosing the best machine learning method for identifying malicious from legitimate websites. One of the major goals is to train machine learning models to find and prevent phishing websites using the dataset. Various models' levels of performance are evaluated and contrasted. The proposed system outperforms state-of-the-art models and demonstrates the importance of hybrid URL features in phishing website detection.

Keywords: *Cybercrime, Phishing, Legitimate websites, URLs, Machine learning approaches, Detection*

1. INTRODUCTION

Phishing involves sending malicious URLs or impersonating trusted people via email or other methods to steal login passwords and payment card information. The victim receives a perceived authentic message from known connections or organizations. The message may contain malicious links, or software, or direct the user to a forged website imitating popular websites. Victims may fall for tricks that deceive them into giving over highly confidential data such as account IDs, login credentials, and credit card numbers. Among cyber attackers, phishing is the most prevalent sort of attack. Phishing assaults are common because victims struggle to understand web applications, computer networks, and technology, making them vulnerable to being misled or spoofed. Phishing unwary consumers into clicking on fake websites for prizes and offers is easier than targeting computer defense systems. The malicious website emulates the organization's emblems and copyrighted content, giving it a real appearance. Individuals and businesses alike can suffer serious reputational and financial setbacks when users fall victim to phishing websites. Malicious PDF or Word attachments are a common feature of phishing emails. When you open a malicious document, it will install malware on your computer. Phishing emails sent from compromised email accounts are easier for cybercriminals to implement than changing the SMTP text message headers.

In recent years, internet usage has increased, leading to increased online transactions,

information sharing, and e-commerce. The rise of the Internet led to the emergence of cybercrime. Cybercriminals often employ phishing to steal information, among other methods. Phishing can take several forms, such as vishing, spear phishing, whaling, and email phishing. Phishing was originally mentioned in 1990 as a method for stealing passwords. Phishing attempts have increased in frequency in recent years. The same is depicted in Figure 1. URL phishing is an example of an attack. A URL is a website address that identifies its location and access method on a network. Accessing the URL connects to the server database, which saves website details and presents them on a webpage. URLs can be dangerous or benign. While benign URLs are safe and secure, URL phishing makes use of malicious URLs. Hackers create fake websites that mimic the actual one at the exact URL. The user's credentials are entered inadvertently when they see the URL as an advertisement on other websites. Another method involves delivering malicious URLs via email, which download viruses when opened, allowing fraudsters to access user data and commit crimes. The goal is to identify dangerous and safe URLs, the features need to be extracted from them. To identify malicious URLs, extract features and compare them to determine if they are malicious or benign.

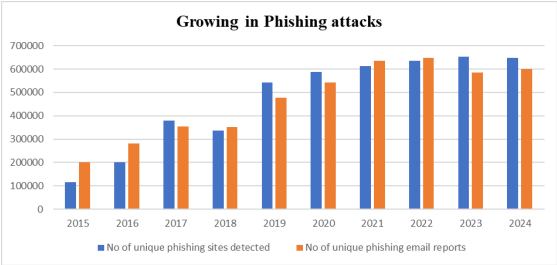

Figure 1. Growth in Phishing attacks over the years

1.1. URL phishing attacks

URL Phishing [1]: Cybercriminals infect targets with URL links. Social individuals are more inclined to accept friend invitations via links, and give contact information, such as email. Attacks sometimes use email or SMS, hidden links, tiny URLs, or misspelled URLs as conduits.

Spear Phishing [1] involves emails with malicious URLs that contain personal information about the target. The email may contain the receiver's name, title, business, social network, and other personal details. The rise of commercial and personal websites, along with social media, allows cybercriminals to gather information and create convincing emails.

Deceptive Phishing [1] is a common phishing attack where cybercriminals impersonate popular entities to steal private data such as usernames, passwords, financial information, and credit card numbers. This approach lacks sophistication due to a lack of personalization and customization for individuals. For instance, when mass emails with phishing URLs are sent to big users, cybercriminals expect people to click them and check malovent URLs or download virus-affected attachments. This sort of phishing involves deceit and impersonation. This type of email often induces panic and haste, prompting victims to reveal vital information. Emails with urgent subject lines, like "Your account has been hacked, change your password immediately!" or "Your bill is overdue-pay immediately or pay fine!", might harm users if they open or visit the URLs.

Phishing a "whale" (top-level executive) involves targeting corporate executives like CEOs or top-level managers [1]. This method of phishing steals CEO secrets and impersonates them. This attack can damage a company's finances, market value, and reputation.

A blacklist database that can be used to avoid being harmed by phishing is provided by security threat intelligence firms that identify and broadcast malicious web URLs or IP addresses. To get sensitive information or passwords, attackers utilize phishing. To do this, we build imitations of our websites in such a way that a user would mistakenly believe they are accessing the legitimate site when they click on the link and enter their credentials.

"Phishing URL Detection, PUD" is a concept of the proposed method that aims to assist users in recognizing phishing websites by analyzing their characteristics using machine learning algorithms. These methods protect user passwords and sensitive data from attackers. The method offers an advanced detection technology that automatically scans websites for hazardous information. This technique is designed for a blacklist provider to automatically create and update a blacklist of hazardous URLs. The system has many features that reflect vital webpage information or behavior that criminals can't hide. The primary goal of the proposed PUD method is to construct a machine learning classifier that can differentiate between legitimate and fraudulent websites based solely on the information provided by the websites themselves. Machine learning models and deep neural networks will be

trained on the dataset to detect whether websites are phishing. The features based on URLs and website content are supplied by the dataset, which includes both malicious and benign URLs.

. Outline of this article: The second section summarizes pertinent studies. The proposed research solutions and their operations are described in Section III. Section IV describes the method's experimental setup and findings. Results and performance discussions are under Section V. Section VI presents the conclusions of the study with future directions.

1.2. Research Objectives:

- To develop a machine learning-based model using hybrid URL features.
- To evaluate multiple ML algorithms on phishing detection tasks.
- To identify the best performing model through comparative experimentation.

2. LITERATURE REVIEW

Currently, the majority of individuals have fallen for identity theft scams and have unwittingly provided hackers with sensitive information. Through the use of deception, consumers are encouraged to provide sensitive information. Numerous prohibited websites have been created masquerading as legitimate ones. Examples include login credentials, financial information, email addresses, and so forth. There was a record amount of phishing activity in early 2021 compared to when the business started measuring it in 2004. The number of phishing attempts recorded in 2021 was 6,122,000. That was an increase of 35.0 percent from the previous year. Monthly phishing attempts in 2015 were 1,124. However, in the last three months of 2024, there was a 5,753% increase in the average number of phishing attacks per month, with 92,564 attacks [1]. Approximately 47,324 phishing attacks occur annually, with a top-ten American bank estimating a loss of USD 300 for every hour that a phishing site is live. Phishing is a hard problem to solve permanently [2] since it usually takes advantage of people's negligence or ignorance and also uses networking technologies. There have been numerous efforts to reduce the risk of phishing attacks by teaching end users to identify and avoid malicious URLs [3, 4].

The regular delivery of notifications to end users alerting them about possible phishing dangers is the main mechanism by which these strategies work. However, they still need the consumers' input and understanding of the underlying tech to function properly [5].

The four main types of automatic phishing detection systems available today are heuristics, visual similarity, machine learning, and blacklist and whitelist methods.

2.1. System for Detecting Phishing Attacks Based on a List

Systematically, these tools can detect phishing websites, and rely on two sets of criteria. A whitelist and a blacklist are two terms that describe these two sets of information. In contrast to the blacklist, which includes phishing sites, the whitelist contains legitimate and secure websites. To identify malicious websites that pose as phishers, this research uses the whitelist. Whitelisting is the only way to access certain websites, according to the research. Employing a blacklist is a further tactic. Several studies employ blacklists, especially along with tools such as Google's safe browsing API and PhishNet [6].

If the URL is not on the blacklist, access to the URL is denied in blacklist-based systems. One big problem with these solutions is that the list stops matching if the URL is slightly different. On top of that, zero-day attacks, the most modern kind of attack are undetectable by these protection technologies.

2.2. Heuristic-Based Phishing Detection Systems

An expansion of black and whitelists could be heuristic phishing detection systems [7]. Commonly used heuristics are signatures associated with previously identified phishing attempts. This method searches websites for the specified signatures and then delivers a warning if malicious conduct is identified [8]. The ability of the heuristic approach to identify newly appearing URLs makes it superior to the blacklist and whitelist approaches. Nevertheless, this strategy outperforms the blacklist and whitelist approaches to phishing detection in terms of false positive rate, mainly because heuristic testing is time-consuming and phishing efforts are complicated.

2.3. Methods for Identifying Phishing Based on Visual Similarity

These programs can do their jobs by comparing the visual similarities between different websites. The server side is responsible for classifying

websites as either phishing or not. We use image processing techniques to compare the two datasets. Fake websites that mimic the appearance of legitimate ones are common. But there are nuanced distinctions in terms of beauty. Using image processing methods could make these small changes easier to notice. If two websites are quite similar, then one of them is probably a phishing attempt. Studies that compare and contrast basic commonalities to uncover differences exist, such as this one [9].

2.4. Machine learning-dependent Phishing Detection Systems

Phishing detection systems that use machine learning rely on the classification of certain properties using AI approaches to identify phishing websites. Features are built using a variety of resources, including collections of URLs, domain names, website content, and features. In terms of user safety, its dynamic structure is quite attractive since it helps to identify suspicious behavior on websites. Machine learning technology makes it easy to find phishing websites. At the same time, it can change to fit different types of phishing websites. Collecting highly qualified attributes from phishing URLs and associated websites is crucial to the efficacy of this approach [10]. The underlying classifier, however, will fail to appropriately identify phishing websites if sensitive criteria are selected inaccurately. Machine learning algorithms are susceptible to overfitting when they are trained with data that contains irrelevant or minor features.

In addition to an awareness survey, Hirmani Sharma and colleagues [11] compared phishing detection programs like Netcraft and SpoofGuard. Over 61% of those who took the survey have no idea what phishing detection software is. Using a database that contained both legitimate and phishing websites, each tool was tested extensively. A whopping 90% of the time, the Anti-phishing toolbar gets it right. Issues: On top of that, it deemed some malicious websites as safe although they were not. Multiple machine learning techniques were employed by Logistic regression, decision trees, and random forests are all part of the toolbox of A. Lakshmanarao, M. M. Bala Krishna, and P. Surya Prabhakara Rao [12]. Data used for the tests came from UCI's machine learning repository. Following that, PA1 and PA2 were employed as algorithms for prioritization. They achieved an accuracy rate of 97% by using a fusion classifier and a final fusion model that was determined by priority-based methods. They relied on just one dataset to test their proposed model.

Phishing attack detection was the goal of Dr. G. Ravi Kumar and colleagues, who utilized several machine-learning approaches. To improve the results, they used Natural Language Processing methods. Their Support Vector Machine, along with data preprocessed using NLP approaches, allowed them to attain remarkable accuracy [13]. The phishing attack detection model created by Venkateshwara Rao et al. [14] using decision trees, support vector classifiers, and random forest models was incredibly effective. While Random Forest achieved an accuracy of 80% on their test data set, Support Vector Machine achieved a whopping 91.3%. By utilizing a random forest technique, Amani Alswailem et al. [15] achieved impressive accuracy when they applied various machine learning models to phishing attempts.

Logistic regression classifiers produced the best results when Meenu et.al. [16] used them to forecast phishing emails in comparison to Decision Tree classifiers, Artificial Neural Networks, Logistic Regression, and, Support Vector Machines. A combination of a naive Bayes classifier and an Artificial Neural Network (ANN) yielded an accuracy of 89.3% when used to identify phishing websites, according to Sandeep Kumar et al. [17]. Having previously proposed a strategy for detecting phishing websites using machine learning methods such as Naive Bayes algorithms, they compared ELM (Extreme Machine Learning) to other ML approaches like ANN and found that it achieved the best accuracy rate of 89.3%. Smriti Dangwal and colleagues argued in a research article [18] that enhanced machine learning algorithms for the detection of phishing websites could be built by defining ideal attributes. Eighteen shared traits were identified after comparing two datasets, one with thirty features and the other with forty-eight. Contrary to earlier research, this study found that the random forest model outperformed the alternatives on the 48-feature dataset, but performed worse on the 30-feature dataset. Up to 93% accurate predictions were made by the superior and more robust 13-feature model. Mahajan Maruiyi Vikas, Sawant Purva, and others [19], investigated several phishing assaults on URLs and made use of ELM to identify phishing websites. The attributes of any website visited by an individual are retrieved via its URL, and the results are utilized as test data.

A combination of the three methods—visual similarity, heuristics, and blacklists and whitelists—

was proposed by Vaibhav Patil and colleagues [20]. Decision trees, logistic regression, and random forests were among the machine learning classifiers utilized on 9076 test websites. The Random Forest approach outperformed the others with a 92% accuracy rate. Problems: Only a few unexpected outcomes (both positive and bad) were identified by this system due to a defect.

A survey paper outlining the merits of Machine Learning detection methods was suggested by Ammar Odeh and colleagues [21]. Listed several issues with phishing detection and ML-based solutions, including ML's inadequacy when faced with large data sets and images. The findings proved that machine learning methods are effective in the battle against phishing. To alert the user via email and pop-ups when a website is not legitimate, Asif Iqbal and colleagues [22] developed a system that directly retrieves blacklisted URLs from the browser. Accuracy was not addressed. Problems: Since the administrator can manually copy and paste URLs to filter banned and non-blacklisted ones, there is a lot of human labor involved.

To improve accuracy while decreasing the number of features, Norah Alrumayh and colleagues [23] examined the 36 features. Achieving better accuracy with a minimum amount of features is our primary objective. The maximum accuracy value that a random forest could produce while using all 29 characteristics together was 90%. Sonmez et al. [24] examined 30 characteristics of phishing websites and their categorization tactics using Extreme Machine Learning. Classification is breaking the whole problem down into a predetermined set of subproblems. Neural Networks, SVMs, and Naive Bayes were the classification algorithms employed. To get a higher accuracy of 95.34%, ELM used six separate activation functions. Ram Basnet et al. [25] proposed using ML models to identify phishing attempts. This research trains six different ML models to detect phishing emails using sixteen different criteria. Although both Biassed SVM and Artificial Neural Networks reach an accuracy of 97.38%, Support Vector Machine (SVM) is the most successful approach.

To identify phishing attempts, Kamal et al. [26] suggested utilizing machine learning using only URL-based data. Since domain names were so easily accessible and inexpensive in 2014, phishing attacks rose, says the APWG. The Naive Bayes method is used on the Weka Platform to categorize phishing websites. The ensemble technique can attain an accuracy of 97.08% by mixing different algorithms like Stacking, Bagging, and Boosting with others like Decision Tree, Random Forest, and Naive Bayes. To combat spam emails and malicious software linked to phishing websites, Baykara et al. [27] developed "Anti Phishing Simulator" software. As an example, it offers a URL-based control to stop major issues like the system catching fraudulent emails sent to internal addresses. This leads to word weight determination and the use of Bayesian classification to calculate spam word counts.

As an optimization strategy, Priya et al. [28] suggested using ant colonies. The proposed system first extracts characteristics of phishing websites and then uses the ant colony optimization method to decrease those attributes. Once again, the features of a webpage are classified and minimized using Naive Bayes. In their demonstration of phishing detection, Priyanka et al. [29] used feature extraction based on machine learning. Using the Adaline and Backpropagation algorithms in conjunction with SVM, their ability to detect and classify web pages was improved. With an accuracy percentage of 99.14%, Adaline outperforms SVM. The Adaline network is faster than the Backpropagation network with SVM in terms of total network time. An Extreme Learning Machine classification algorithm was developed by Mustafa Kaytan et al. [30] to detect phishing web pages. Using "Request URL" and "Website Forwarding" as criteria, this study classifies phishing websites. The 10-cross-fold validation method is used to evaluate performance. The highest level of accuracy achieved was 95.33%, while the average was 95.05%. This study also suggested a method to detect dangerous URLs based on HTML properties; specifically, they suggested utilizing the beautiful soup Python package, which parses HTML and XML files, to check if the URL contains any harmful content. Another option is to use string-based algorithms. These algorithms can reprocess URLs such that both malicious and legitimate URLs have a word cloud. However, in this case, the word cloud only contains the most common words in legitimate URLs and their corresponding malicious counterparts. Then, the algorithm can analyze the word clouds to determine which URLs are more malicious. It is possible to use machine learning algorithms to determine the safety of a URL. A literature review is summarised in Table I.

All the above solutions presented fail to address properly the phishing in a diverse environment. This article provides a comprehensive

solution to build a prototype capable of identifying phishing attempts and, if successful, the level of sophistication of such attacks.

While many past studies have used either lexical or content-based features, few have explored hybrid URL structures using a comprehensive set of attributes. Some models rely on heavy computational infrastructure or require deep content analysis. This study addresses these limitations by proposing a lightweight, hybrid-feature-based system using widely accessible machine learning techniques.

3. MATERIALS AND METHODS

The study's objective is to evaluate several ML algorithms for identifying phishing websites. The primary goal of the suggested approach is to develop a model that can determine the likelihood of a website being a phishing site and the severity of the attack. Websites related to Phishing can be detected by several qualities and characteristics, such as spelling mistakes, long URLs, customization, prefixes, and suffixes. These attributes are extracted from input web pages using a variety of approaches. The results of the proposed method will pave the way for further research into methods for identifying and preventing phishing websites.

Figure 2 depicts the methodology of the presented method. The first move is to gather all the URLs, both legitimate and malicious. The lexical properties of these URLs are retrieved. The feature selection method narrows the search to the most relevant attributes. This method ranks attributes according to their contribution to phishing and non-phishing threat detection. The efficiency of several classification methods is then tested for different amounts of URLs. The key steps of the suggested method " PUD" are described:

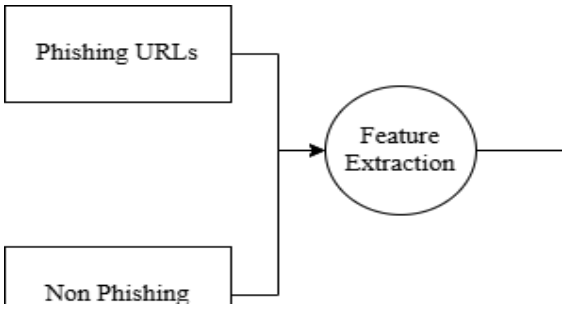

Figure 2. The framework of the suggested methods

3.1. Data Pre-processing

Data often conjures images of enormous databases with many rows and columns. Even while this happens frequently, it doesn't mean it always does; extra data could arrive in several forms: Pics, Tabs, and Organised Tabs. The only data that computers can understand is binary data, such as 1s and 0s. In short, it could be unrealistic to expect a model based on machine learning to learn only by viewing a presentation of all our data. Data as part of any Computer Learning process, pre-processing involves encoding the data so that the machine can better grasp it. Some of the sequential sub-processes that comprise it are data cleaning, data transformation, data reduction, data quality assessment, and so on.

3.2. Extraction

Using a Python application, URL attributes are able to be retrieved. The following are the attributes that can able to extracted to identify phishing URLs.

3.2.1 IP address presence

It looks for an IP address in the URL. For certain URLs, it is feasible to use IP addresses rather than domain names. One well-known method of stealing sensitive data is to use an IP address in URLs instead of a domain name. This attribute can be adjusted to either 0 (legal) or 1 (phishing) depending on whether an IP address is present in the domain component of the URL.

3.2.2 @ Symbol presence

All URLs are validated to ensure they contain the '@' symbol. After the "@" sign, the browser starts to pay attention to the URL, and the real address is typically found after the "@" sign. The feature is set to 1 (phishing) if the URL contains the '@' symbol and to 0 (legal) otherwise.

3.2.3 Length of URL

Returns the URL length. Senders of phishing emails often use long URLs to hide any potentially malicious content in the address bar. Phishing URLs are defined in this project as those with a length of 54 characters or more. One (phishing) or zero (legal) is the feature's value when the URL length exceeds 54 characters.

3.2.4 The URL's depth

The depth of the URL is set. The number of subpages in the given URL is determined by this functionality, which is based on the '/'. A value is assigned to the feature based on the URL.

3.2.5 "//" Redirection in URL

Verifying if the URL has the "//" symbol is done. Any website that has the character "//" in its URL path will redirect the user to an altogether different page. The computation of the "//" in a URL's location is done. To properly format URLs beginning with "HTTP," we discovered that the "//" should go into position six. However, if the URL uses "HTTPS," the "//" should be at position seven. When the "//" character appears in the URL anywhere other than after the protocol, the attribute is set to 1 (phishing), and 0 (legal) otherwise.

3.2.6 Domain name in HTTP/HTTPS

The presence of "http/https" in the domain portion of the URL has been verified. Criminals can trick people into accessing malicious websites by adding the "HTTPS" token to the domain part of the URL. This attribute is set to 1 (phishing) if the domain section of the URL contains "http/https," and to 0 (legal) otherwise.

3.2.7 "TinyURL" - URL mini Services

A URL can be "shortened" to a more manageable length via URL shortening, all while still directing users to the desired destination on the "World Wide Web." This is achieved by the use of the "HTTP Redirect" mechanism, which allows shorter domain names to redirect traffic to longer URLs. There are two possible values for this attribute: 1 (phishing) and 0 (legal) based on whether the URL uses Shortening Services or not.

3.2.8 Domain prefix or suffix with a period (-)

The feature is set to 0 unless the domain name contains a dash (-). Real URLs rarely feature the dash symbol. To make their fraudulent websites appear more official, phishers add a dash (-) to the domain name. For instance, while the official Amazon website is www.onlineamazon.com, phishers can set up a phony site at http://www.online-amazon.com to trick people.so that it appears to be a genuine website. For instance, while the official Amazon website is http://www.onlineamazon.com, phishers can set up a phony site at http://www.online-amazon.com to trick people.

3.2.9 DNS Record

Regarding phishing websites, the WHOIS database does not contain any entries for the hostname or does not recognize the stated identity. If the DNS record is empty or not discovered, the value of this characteristic is set to 1 (phishing), and otherwise, it is set to 0 (legal).

3.2.10 Web Traffic

By tallying the total number of visitors and the amount of pages they view, this function determines the website's popularity. The short lifespan of phishing sites means that they can go unnoticed by the Alexa database (Alexa the Web Information Company., 1996). According to our findings, genuine websites were able to achieve rankings in the top 100,000 even in the most challenging circumstances. According to Alexa, it is also deemed as "Phishing" if the domain is not recognized or has no traffic at all. If the domain rank is less than 100,000, this characteristic is set to 1 (phishing), and 0 (legal) otherwise.

3.2.11 Domain Age

Searching the WHOIS database will yield this data. Most phishing websites have a very limited lifespan. A lawful domain is considered to be at least 12 months old for this method. The definition of age is the difference between the creation and expiration timestamps. This attribute's value is 1 (phishing) if the domain is more than 12 months old, and 0 (legal) otherwise.

3.2.12 Decline in Domain Life

Another source that could supply this data is the WHOIS database. The remaining domain time for this feature is determined by subtracting the termination time from the current time. Half a dozen months or less is considered the valid domain's finish time for this project. Domains with expiration dates longer than six months have a value of one (phishing), whereas those with shorter expiration dates have no value (legal).

3.2.13 IFrame Redirection

The IFrame element in HTML lets the user insert a new window within the current one. Phishers can use the "iframe" tag to make the frame invisible and borderless. Here, phishers use the "frame border" feature to make the browser draw a border. One (phishing) or zero (legal) is assigned to this feature based on whether the iframe is empty or there is no response.

3.2.14 Personalisation of the Status Bar

Web users are vulnerable to malicious URLs shown in the status bar due to JavaScript. To access this feature, you need to look for the "onMouseOver" event in the page's source code and see if it modifies the status bar. Discovered when

the answer is empty or when "onMouseOver" is found, this feature is set to 1 for phishing and 0 for legitimate.

3.2.15 Preventing Right Click

To prevent users from being able to view or store a webpage's source code, phishers utilize JavaScript to restrict the right-click function. Thinking about this function in the same way as "Hiding the Link using onMouseOver." However, to ascertain whether the right-click functionality of this feature is disabled, we will continue to search for the event "event.button==2" in the webpage's source code. If the response is empty or the onMouseOver attribute cannot be found, the feature's value is set to 1 (phishing) or 0 (legal).

3.2.16 Website Forwarding

An indicator of a phishing website is the frequency with which it redirects users away from the original. One route was found for authentic websites in our sample. In contrast, at least four phishing websites have been routed utilizing this functionality.

3.3 Machine Learning Models Employed

Extracting these attributes from input web pages is done using a variety of approaches. Notably, this method's results will provide a path for further research into phishing website prediction and detection. The following are examples of top-tier supervised machine learning models for classification that will be utilized to develop the ML model.

3.3.1 Decision Tree Classifier

A decision tree classifier is a useful tool for regression and classification jobs that involve selecting a choice, like if/else statements. A perfect choice maker, this one can get to the right decision in record time.

3.3.2 Random Forest Classifiers

Its regression and classification capabilities make it a popular machine-learning method. Although individual trees in a random forest may provide respectable predictions, the underlying principle is that they will likely overfit certain data. They don't necessitate data scalability, are frequently effective with little to no parameter tweaking, and pack a mighty punch.

3.3.3 Multilayer Perceptron

A feedforward neural network is a multilayer perceptron. They evaluate many options for each stage simultaneously and choose the best one.

3.3.4 XGBoost Classifier

XGBoost uses decision trees as part of an ensemble Machine Learning approach based on gradient boosting. Like other classification or regression algorithms, XGBoost is built to be fast and perform well. Decision trees will have gradient boosting applied to them.

3.3.5 Support Vector Machines

One kind of technique that is suitable for both classification and regression is the support vector machine, which is sometimes called a support vector network. After each new output is analyzed, the imported training data set will be split into two groups.

3.4 Methodology

The proposed method not only retrieves a dataset from a database that contains both legitimate and malicious URLs, but it also pre-processes the data. These four types of URL attributes—domain-based, address-based, anomalous-based, and HTML, JavaScript elements—are shown in Figure 3 and are utilized to identify phishing websites. All of the URL attributes get new values after processing and retrieving some of their features with the data. Before analyzing a URL, a machine learning system establishes a range and threshold for its attributes. By doing so, you can tell if the URL is genuine or a phishing scam.

The following steps are used in the suggested method:

- Data Extraction & Validation: Extraction of Data is the procedure of bringing information from one source to another, whether that's in the cloud, on-premises, or some combination of the two. Several techniques, some of which are quite sophisticated and others of which are more commonly executed by hand, are employed to achieve the current result. The ETL process (Extraction, Transformation, and Loading), is typically the first stage unless the knowledge is being extracted solely for repository purposes. This shows that information is almost constantly subjected to further processing upon initial retrieval to make it suitable for further analysis.

- Identify the necessary characteristics: To identify the features that this project must have.
- Extracting features from the obtained dataset: Feature extraction is the process of transforming knowledge, data, or information into numerical alternatives that may be processed while preserving the original data set's understanding. Compared to using machine learning on the data, it produces better outcomes.
- Use several deep neural network and machine learning techniques on the dataset, such as XGBoost, decision tree classifiers (DT), support vector machines (SVMs), multilayer perceptrons (MLPs), and random forests (RFs). Applying metrics for correctness to the evaluation of the results.
- Find the best model by comparing the acquired results from training models.
- There are five methods available for checking the validity of a URL.
- URLs were collected in an unorganized manner from the PhishTank website and the "University of New Brunswick".
- The unstructured input is transformed into nine features during the pre-processing phase.
- An IP address, length, phishing phrase, number of dots, number of slashes, suspicious characters, and HTTP status can all be used to identify a URL.
- Afterwards, each piece of information, including the paired (0,1), is added to a well-organized dataset, which is subsequently used by the different classifiers.
- Next, five separate classifiers' performance is trained and evaluated: Random Forest, Decision Tree, SVM, XGBoost, and the Multilayer Perceptron method.

3.4 Dataset Description

3.4.1 Legitimate URLs

All of the legitimate URLs came from free datasets hosted by the University of New Brunswick. All sorts of malicious, spammy, phishing, and defacement URLs are included in this dataset. In this study, we are examining the benign URL dataset among these types. From this collection, over 5,000 randomly selected valid URLs will be retrieved. All characteristics that have their origins in a valid URL are listed in Table 2.

3.4.2 Phishing URLs

The open-source PhishTank tool collected the phishing URLs. It updates phishing URLs hourly in CSV, JSON, and other forms. We'll get around 5000 random phishing URLs from this dataset. All phishing URL attributes are shown in Table 3.

There have been reports of various phishing websites that are supposedly inactive, even though the majority of them are only meant for short usage. The dataset must undergo a filtration phase to ensure it is up-to-date. A plethora of phishing websites become accessible after this screening. This analysis makes use of a fresh dataset.

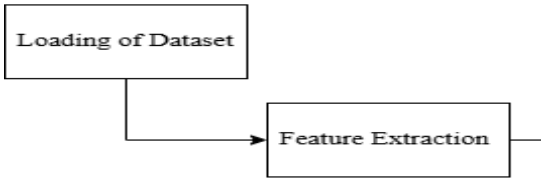

Figure 3. Process flow of the suggested method

4. EXPERIMENTATION

The first step in starting the experimentation was to import all of the packages indicated in the diagram. Creating data frames, cleaning and pre-processing data, and plotting graphs when needed are the primary uses of the imported software. The same is depicted in Figure 4.

```
[1] #importing basic packages
    import pandas as pd
    import numpy as np
    import seaborn as sns
    import matplotlib.pyplot as plt
```

Figure 4. Importing packages

The final dataset created after all the feature extractions is named "urldata.csv," and imported here, as seen in Figure 5. Panda's library of Python is utilized to upload the dataset. In the future, this dataset will undergo cleaning and pre-processing before being split evenly between training and testing.

Journal of Theoretical and Applied Information Technology
31st August 2025. Vol.103. No.16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

```
#Loading the data
data0 = pd.read_csv('urldata.csv')
data0.head()
```

Figure 5. Loading dataset using pandas

The features of the dataset have been gathered and listed along with their names so that you can get a sense of what is being used. These characteristics are crucial for the model to identify fraudulent URLs. You can see this in action in Figure 6.

```
#Listing the features of the dataset
data0.columns

Index(['Domain', 'Have_IP', 'Have_At', 'URL_Length', 'URL_Depth',
       'Redirection', 'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record',
       'Web_Traffic', 'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over',
       'Right_Click', 'Web_Forwards', 'Label'],
      dtype='object')
```

Figure 6. Extracted Feature of the Dataset

Figure 7 presents that the data is divided into a 90:10 split for testing and training. Datasets are split into two subsets as part of the procedure. When fitting a model, the first subset used is the training dataset. Instead of using the second subset for training, the model takes it as input and uses it to make predictions and check if they match the values predicted. "Test dataset" is the name given to the second dataset. To get the most out of our dataset, the suggested strategy trains on 90% of it and tests on 10%.

```
# Splitting the dataset into train and test sets: 90-10 split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size = 0.1, random_state = 12)
X_train.shape, X_test.shape

((6201, 16), (689, 16))
```

Figure 7. Extracted Feature of the Dataset

In order to compare the models' test and train accuracy, the final step is to initialize them. This is done after the data is divided into training and testing sets.

5. RESULTS AND DISCUSSIONS

The proposed method, PUD, and data model will detect phishing website URLs. PhishTank, an open-source application, randomly chose 5000 URLs. Phishing URLs are updated hourly in CSV, JSON, and other formats by this service. The input URL is either a phishing attempt (1) or a real website (0), making classification difficult in this dataset. The approach takes datasets of legitimate and fraudulent websites from open-source platforms, extracts the necessary features from the URL database, and then analyses and pre-processes the dataset using EDA methods. We will use the dataset to construct two sets: one for training and one for testing. Once the dataset has been partitioned into training and testing sets, machine learning and deep neural network approaches are employed. Finally, the proposed method displays accuracy metrics evaluation results. In the end, compare all modules to choose the most accurate phishing detection algorithm and classify the dataset as phishing or genuine to get the right website search results.

In a quest to find the optimal balance between powerful feature combinations that would minimize computation time and maximize performance, all seventeen features are thoroughly examined. Since the majority of the data consists of integers, with the non-inclusion of two features, 'URL_Depth, and 'Domain' which were removed. These features are completely irrelevant when it comes to training a machine-learning model.

This section presents the outcomes of the suggested model together with the other approaches that were used as samples. In order to ensure transparency and accuracy, each classifier was constructed with an identical performance Metric.

5.1 Decision Tree Classifier

Figure 8 shows the computation of the Decision Tree classifier's accuracy, as well as the results of the test and train accuracy analyses.

```
#computing the accuracy of the model performance
acc_train_tree = accuracy_score(y_train,y_train_tree)
acc_test_tree = accuracy_score(y_test,y_test_tree)

print("Decision Tree: Accuracy on training Data: {:.3f}".format(acc_train_tree))
print("Decision Tree: Accuracy on test Data: {:.3f}".format(acc_test_tree))

Decision Tree: Accuracy on training Data: 0.891
Decision Tree: Accuracy on test Data: 0.894
```

Figure 8. Decision Tree Classifier Accuracy on Test and Train Data.

To know which attributes are most heavily utilized by the Decision Tree model. We have included code that generates a self-explanatory graph and helps you determine the value of the retrieved data. The code that ranks the features in the Decision Tree method is shown in Figure 9.

Journal of Theoretical and Applied Information Technology
31st August 2025. Vol.103. No.16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

```
#checking the feature improtance in the model
plt.figure(figsize=(9,7))
n_features = X_train.shape[1]
plt.barh(range(n_features), tree.feature_importances_, align='center')
plt.yticks(np.arange(n_features), X_train.columns)
plt.xlabel("Feature importance")
plt.ylabel("Feature")
plt.show()
```

Figure 9. Importance of features in Decision Tree Method.

Figure 10 is a graph presentation of the decision tree model's significance.

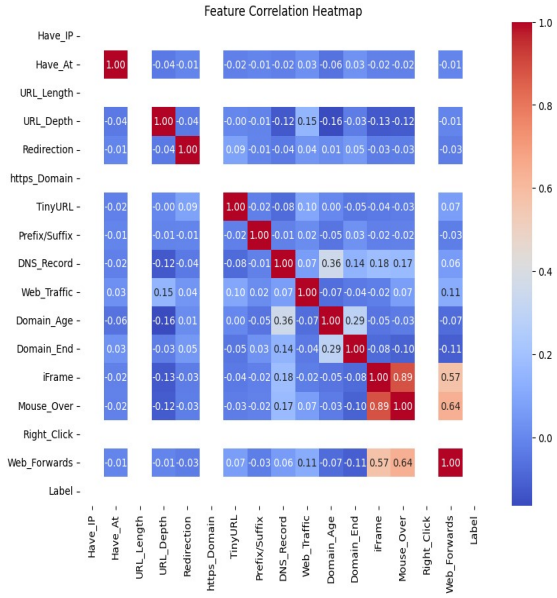

Figure 10. Decision Tree Classifier feature importance graph.

5.2 Random Forest Classifier

Figure 11 shows the code utilized to calculate the Random Forest classifier's accuracy, as well as the results of the test and train accuracy analyses.

```
#computing the accuracy of the model performance
acc_train_forest = accuracy_score(y_train,y_train_forest)
acc_test_forest = accuracy_score(y_test,y_test_forest)

print("Random forest: Accuracy on training Data: {:.3f}".format(acc_train_forest))
print("Random forest: Accuracy on test Data: {:.3f}".format(acc_test_forest))

Random forest: Accuracy on training Data: 0.884
Random forest: Accuracy on test Data: 0.887
```

Figure 11. The Performance of Random Forest Classifiers on Real-World Datasets.

Likewise, it would be helpful to know which properties the Random Forest model uses the most. A self-explanatory graph is generated for calculating the relevance of the extracted features. The same is depicted in Figure 12 below which prioritizes features in the Random Forest Method. Figure 13 is a graph representation of the Random Forest Model's significance.

```
#checking the feature improtance in the model
plt.figure(figsize=(9,7))
n_features = X_train.shape[1]
plt.barh(range(n_features), forest.feature_importances_, align='center')
plt.yticks(np.arange(n_features), X_train.columns)
plt.xlabel("Feature importance")
plt.ylabel("Feature")
plt.show()
```

Figure 12. Random Forest Method: Features and Their Importance.

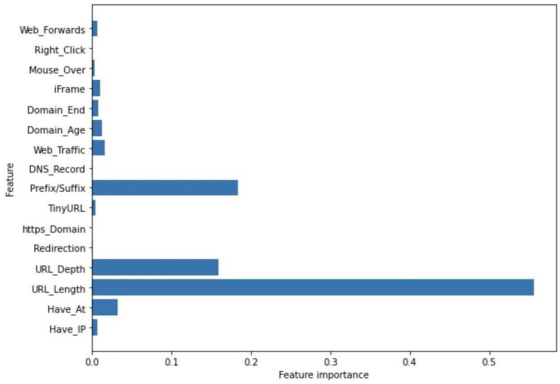

Figure 13. Priority Graph for Random Forest Classifier Features.

5.3 Multilayer Perceptron Classifier

Figure 14 displays the computation of the Multilayer Perceptron classifier's accuracy, as well as the results of the test and train accuracy runs.

```
#computing the accuracy of the model performance
acc_train_mlp = accuracy_score(y_train,y_train_mlp)
acc_test_mlp = accuracy_score(y_test,y_test_mlp)

print("Multilayer Perceptrons: Accuracy on training Data: {:.3f}".format(acc_train_mlp))
print("Multilayer Perceptrons: Accuracy on test Data: {:.3f}".format(acc_test_mlp))

Multilayer Perceptrons: Accuracy on training Data: 0.902
Multilayer Perceptrons: Accuracy on test Data: 0.911
```

Figure 14. The Performance of Multilayer Perceptron's on Real-World Datasets.

5.4 XGBoost Classifier

Journal of Theoretical and Applied Information Technology
31st August 2025. Vol.103. No.16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 15 shows the test and train accuracy output and XGBoost classifier accuracy computation.

```
#computing the accuracy of the model performance
acc_train_xgb = accuracy_score(y_train,y_train_xgb)
acc_test_xgb = accuracy_score(y_test,y_test_xgb)

print("XGBoost: Accuracy on training Data: {:.3f}".format(acc_train_xgb))
print("XGBoost : Accuracy on test Data: {:.3f}".format(acc_test_xgb))

XGBoost: Accuracy on training Data: 0.917
XGBoost : Accuracy on test Data: 0.917
```

Figure 15. Testing and Training Data Accuracy of the XGBoost Classifier.

5.5 Support Vector Machine Model

Figure 16 shows the calculation of the Support Vector Machine classifier's accuracy, as well as the results of the test and train accuracy analyses.

```
#computing the accuracy of the model performance
acc_train_svm = accuracy_score(y_train,y_train_svm)
acc_test_svm = accuracy_score(y_test,y_test_svm)

print("SVM: Accuracy on training Data: {:.3f}".format(acc_train_svm))
print("SVM : Accuracy on test Data: {:.3f}".format(acc_test_svm))

SVM: Accuracy on training Data: 0.882
SVM : Accuracy on test Data: 0.882
```

Figure 16. The efficiency of Support Vector Machine Classifiers on Real-World Data Sets.

The results of the models are sorted and presented below according to their accuracy. Table 3 below shows the outcomes of the results compared to certain classifiers: XGBoost, Multilayer Perceptrons, Decision Tree, Random Forest, and SVM.

Table 4. Phishing URL Dataset Structure

Machine Learning Modes	Train Accuracy	Test Accuracy
XGBoost	0.936	0.942
Multilayer Perceptrons	0.913	0.926
Decision Tree	0.897	0.898
Random Forest	0.887	0.891
Support vector machine	0.884	0.886

As demonstrated in the comparison table 4 above, the XGBoost Classifier outperforms Multilayer Perceptron with this dataset.

To make it easier to grasp how the model's performance differs. Using a graph, we have demonstrated how each model performed. Figure 17 demonstrates that out of all the algorithms, the suggested model had the highest accuracy, whereas Random Forest, Decision Tree, and Support Vector Machine had the lowest. In terms of overall reliability, the experimental findings show that the XGBoost algorithm stands ahead of the competition.

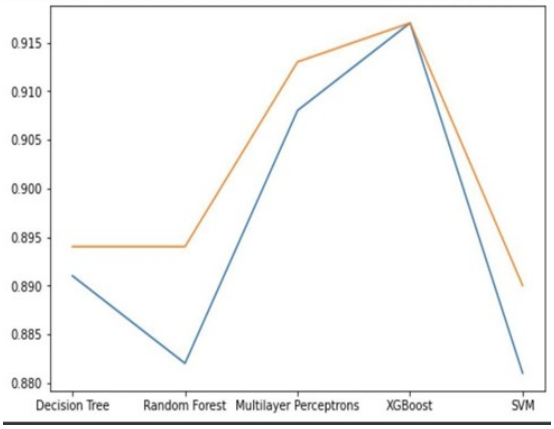

Figure 17. Validity Graph for Model Comparisons.

When compared to more conventional methods of gradient boosting, XGBOOST offers numerous benefits. Among the many advantages are improved regularisation capabilities, which lessen the likelihood of overfitting; rapid creation of trees, which improves speed and performance; adaptability, which allows for various optimization goals and evaluation criteria; and built-in procedures for dealing with missing variables.
The PUD model differs from prior works in three key ways:

- Uses a comprehensive hybrid feature set from lexical, host-based, and behavioral indicators.
- Benchmarks multiple ML models fairly on the same dataset.
- Achieves higher accuracy (94.2%) than prior models tested on similar datasets, such as the 92% accuracy reported by Patil et al. using Random Forest [20].

5.6 Discussions

- On this dataset, Support Vector Machines perform poorly with linearly separable data. While the data does become separable, it is still not directly linearly separable, and support vector machines (SVMs) struggle to learn from it. Consequently, an accuracy level of 88.6% is provided.

- When compared to other models, the Random Forest Classifier demonstrates comparable performance on the provided dataset, albeit with a lower accuracy of 89.1% and a greater false positive rate.
- Decision trees outperform support vector machines (SVMs) and random forests (RFs) on the provided data following pruning, with an accuracy of 89.8 percent, since they can identify patterns that are not linearly separable.
- Similar to the human brain, the multi-layered perceptron is made up of neurons that are connected and share information. An identifier is given to every neuron. With an accuracy of 92.6%, this model was the second most accurate.
- XGBoost employs gradient boosting to decrease mistakes; it is an ensemble model that is decision tree-based. With its 94.2% accuracy rate and methods that are comparable to gradient descent, it is ideal for categorical data.
- Our model achieves 94.2% accuracy, higher than the typical 88–92% range.
- Hybrid features led to lower false positives.
- XGBoost showed better generalization than decision tree and SVM-based models.

6. CONCLUSION

The proliferation of phishing attacks in recent years is directly attributable to the impact of new networking technologies on both traditional web apps and the ever-evolving digital and networking tools. When it comes to security, most attackers aim for system weaknesses, but phishing makes use of human end-user shortcomings. Therefore, organizations' principal line of defense is to educate their staff about this type of attack. Additionally, security teams have the option to acquire additional protection mechanisms that can be utilized for either assisting users with their choices or preventing servers from being compromised.

In this study, a phishing detection system, PUD, using machine learning techniques is developed. To test the proposed systems, several current datasets are utilized. All five ML methods were trained and tested on the same dataset so that the comparison would be fair. The results showed that the model is robust, outperforming the other models studied when the XGBoost algorithm was used to detect phishing websites. Prediction performance for identifying phishing websites is improved by 94.2% using the XGBoost approach. The XGBoost method outperforms every other model that was examined, including Random Forest, Decision Tree, Multilayer Perceptron, and Support Vector Machines.

The proposed method achieved very high detection accuracy, according to the experimental results, when compared to the relevant prior study. These promising outcomes show that phishing can still be effectively tackled by Machine Learning in contrast to the majority of current anti-phishing strategies. The length of time required for feature extraction and training is one of the hurdles to be overcome. The model also has the limitation of not being able to determine if the URL is active or not, hence, the URLs should be checked whether it is active before detection to make sure it works.

The proposed PUD model successfully achieved the study's objectives by building a hybrid-feature detection system, testing five different classifiers, and identifying XGBoost as the most effective model. Future work can explore integration of webpage content and user behavior, improve generalization to multilingual phishing, and deploy the model in browser-based environments for real-time detection.

REFERENCES:

[1]. Xu, Yingying, Guangxuan Chen, Qiang Liu, Wanpeng Xu, Lei Zhang, Jiajian Wu, and Xiaoshi Fan. "A Phishing Website Detection and Recognition Method Based on Naive Bayes." In 2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC), vol. 6, pp. 1557-1562. IEEE, 2022.

[2]. M. Khonji, Y. Iraqi, and A. Jones, ‘‘Phishing detection: A literature survey,’’ IEEE Commun. Surveys Tuts., vol. 15, no. 4, pp. 2091–2121, 4th Quart., 2013.

[3]. A. Acquisti, I. Adjerid, R. Balebako, L. Brandimarte, L. F. Cranor, S. Komanduri, P. G. Leon, N. Sadeh, F. Schaub, M. Sleeper, Y. Wang, and S. Wilson, ‘‘Nudges for privacy and security: Understanding and assisting users’ choices online,’’ ACM Computing Surv., vol. 50, no. 3, 2017, Art. no. 44.

[4]. M. M. Moreno-Fernández, F. Blanco, P. Garaizar, and H. Matute, ‘‘Fishing for phishers. Improving Internet users’ sensitivity to visual deception cues to prevent electronic fraud,’’ Comput. Hum. Behav., vol. 69, pp. 421–436, Apr. 2017.

[5]. M. Junger, L. Montoya, and F.-J. Overink, ‘‘Priming and warnings are not effective to

prevent social engineering attacks,'' Comput. Hum. Behav., vol. 66, pp. 75–87, Jan. 2017.

[6]. M. Sharifi and S. H. Siadati, "A phishing sites blacklist generator," 2008 IEEE/ACS International Conference on Computer Systems and Applications, pp. 840–843, 2008.

[7]. F. Vanhoenshoven, G. Nápoles, R. Falcon, K. Vanhoof, and M. Köppen, ''Detecting malicious URLs using machine learning techniques,'' in Proc. IEEE Symp. Ser. Comput. Intell. (SSCI), Dec. 2016, pp. 1–8.

[8]. J. Saxe, R. Harang, C. Wild, and H. Sanders, ''A deep learning approach to fast, format-agnostic detection of malicious Web content,'' in Proc. IEEE Symp. Secure. Privacy Workshops (SPW), San Francisco, CA, USA, Aug. 2018, pp. 8–14

[9]. L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, and X. Deng, "Detection of phishing webpages based on visual similarity," Special interest tracks and posters of the 14th international conference on World Wide Web - WWW 05, pp. 1060-1061, 2005.

[10]. G. Xiang, J. Hong, C. P. Rosé, and L. Cranor, ''CANTINA+: A feature-rich machine learning framework for detecting phishing Web sites,'' ACM Trans. Inf. Syst. Secure., vol. 14, no. 2, 2011, Art. no. 21.

[11]. Sharma, Himani, Er Meenakshi, and Sandeep Kaur Bhatia. "A comparative analysis and awareness survey of phishing detection tools." In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pp. 1437-1442. IEEE, 2017.

[12]. Lakshmanarao, A., Rao, P.S.P. and Krishna, M.B., 2021, March. Phishing website detection using novel machine learning fusion approach. In the 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)(pp. 1164-1169). IEEE.

[13]. Kumar, G.R., Gunasekaran, S. and AS, V., 2018. URL Phishing Data Analysis and Detecting Phishing Attacks using Machine Learning in NLP. International Journal of Engineering Applied Sciences and Technology (IJEAST), 3(8), pp.70-75.

[14]. Rao, K. Venkateswara. "An Approach for Detecting Phishing Attacks Using Machine Learning Techniques." Journal of Critical Reviews 7, no. 18 (2020): 321-324.

[15]. Alswailem, Amani, Bashayr Abdullah, Norah Alrumayh, and Aram Alsedrani. "Detecting phishing websites using machine learning." In 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS), pp. 1-6. IEEE, 2019.

[16]. Meenu, Sunil Godara. "Phishing detection using machine learning techniques." Int J Eng Adv Technol 9, no. 2 (2019).

[17]. Satapathy, Sandeep Kumar, Shruti Mishra, Pradeep Kumar Mallick, Lavanya Badiginchala, Ravali Reddy Gudur, and Siri Chandana Guttha. "Classification of features for detecting phishing websites based on machine learning techniques." International Journal of Innovative Technology and Exploring Engineering, volume8 (8S2) (2019): 425-430.

[18]. Dangwal, Smriti, and Arghir-Nicolae Moldovan. "Feature Selection for Machine Learning-based Phishing Websites Detection." In 2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), pp. 1-6. IEEE, 2021.

[19]. Vilas, Mahajan Mayuri, Kakade Prachi Ghansham, Sawant Purva Jaypralash, and Pawar Shila. "Detection of Phishing Website Using Machine Learning Approach." In 2019 4th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT), pp. 384-389. IEEE, 2019.

[20]. Patil, Vaibhav, Pritesh Thakkar, Chirag Shah, Tushar Bhat, and S. P. Godse. "Detection and prevention of phishing websites using machine learning approach." In 2018 Fourth International Conference on Computing Communication Control and automation (ICCUBEA), pp. 1-5. Ieee, 2018.

[21]. Odeh, Ammar, Ismail Keshta, and Eman Abdelfattah. "Machine Learning Techniques for Detection of Website Phishing: A Review for Promises and Challenges." In 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0813-0818. IEEE, 2021.

[22]. Alkawaz, Mohammed Hazim, Stephanie Joanne Steven, and Asif Iqbal Hajamydeen. "Detecting phishing website using machine learning." In 2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA), pp. 111-114. IEEE, 2020.

[23]. Alswailem, Amani, Bashayr Abdullah, Norah Alrumayh, and Aram Alsedrani. "Detecting phishing websites using machine learning." In 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS), pp. 1-6. IEEE, 2019.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[24]. Sönmez, Yasin, Türker Tuncer, Hüseyin Gökal, and Engin Avcı. "Phishing websites feature classification based on extreme learning machine." In 2018 6th International Symposium on Digital Forensic and Security (ISDFS), pp. 1-5. IEEE, 2018.

[25]. Basnet, Ram, Srinivas Mukkamala, and Andrew H. Sung. "Detection of phishing attacks: A machine learning approach." In Soft computing applications in industry, pp. 373-383. Springer, Berlin, Heidelberg, 2008.

[26]. Kamal, Gyan, and Monotosh Manna. "Detection of phishing websites using naïve Bayes algorithms." International Journal of Recent Research and Review 11, no. 4 (2018): 34-38.

[27]. Baykara, Muhammet, and Zahit Ziya Gürel. "Detection of phishing attacks." In 2018 6th International Symposium on Digital Forensic and Security (ISDFS), pp. 1-5. IEEE, 2018.

[28]. Priya, R. "An ideal approach for detection of phishing attacks using naïve Bayes classifier." Int. J. Comput. Trends Technol 40, no. 2, 2016.

[29]. Singh, Priyanka, Yogendra PS Maravi, and Sanjeev Sharma. "Phishing websites detection through supervised learning networks." Computing and Communications Technologies (ICCCT), 2015 International Conference on. IEEE, 2015, pp. 61-65.

[30]. Kaytan, Mustafa, and Davut HANBAY. "Effective classification of phishing web pages based on new rules by using extreme learning machines." Computer Science 2, no. 1 (2017): 15-36.

Table 1 Comparative analysis of existing literature

Ref.	Author	Approach	Strength	Weakness	Scope in Proposed Method
[20]	Vaibhav Patil. et.al	Decision trees, logistic regression, and random forests were among the machine learning classifiers utilized on 9076 test websites.	The Random Forest algorithm outperformed the others with a 92% accuracy rate. Problems:	Because of an issue, this system detected a tiny number of false positive and negative findings.	Machine learning classifiers can be utilized on test websites.
[22]	Asif Iqbal et.al .	Developed a system that directly retrieves blacklisted URLs from the browser.	Balcklistes URL's	Accuracy was not addressed. Since the administrator can manually copy and paste URLs to filter banned and non-blacklisted ones, there is a lot of human labor involved.	Direct retrieving
[23]	Norah Alrumayh et al.	Examined the 36 features. The maximum accuracy value that a random forest could produce while using all 29 characteristics together was 90%.	To reduce feature count without sacrificing accuracy	Difficult to identify offline parameters	Task sequencing
[24]	Sonmez et al.	Examined 30 characteristics of phishing websites and their categorization tactics using Extreme Machine Learning.	Neural Networks, SVMs, and Naive Bayes were the classification algorithms employed.	Classification is breaking the whole problem down into a predetermined set of subproblems.	Features to be an examiner for detecting malicious URLs
[25]	Ram Basnet et al.	This research trains six different ML models to detect phishing emails using sixteen different criteria.	The most effective method is Support Vector Machine (SVM), while both Biassed SVM and Artificial Neural Networks achieve an accuracy of 97.38%.	Only sixteen distinct properties.	Machine learning methods for the detection of phishing emails

Journal of Theoretical and Applied Information Technology
31st August 2025. Vol.103. No.16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2. Legitimate URL Dataset Structure

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End
0	graphicriver.net	0	0	1	1	0	0	0	0	0	1	1	1
1	ecnavi.jp	0	0	1	1	1	0	0	0	0	1	1	1
2	hubpages.com	0	0	1	1	0	0	0	0	0	1	0	1
3	extratorrent.cc	0	0	1	3	0	0	0	0	0	1	0	1
4	icicibank.com	0	0	1	3	0	0	0	0	0	1	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4995	getpocket.com	0	0	1	1	1	0	1	0	0	1	0	1
4996	olx.ro	0	0	1	7	0	0	0	0	0	1	0	0
4997	medium.com	0	1	1	2	0	0	0	0	0	1	0	1
4998	thenextweb.com	0	0	1	6	0	0	0	0	0	1	0	0
4999	smallseotools.com	0	0	1	2	0	0	0	0	0	1	0	1

5000 rows × 18 columns

Table 3. Phishing URL Dataset Structure

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End
0	graphicriver.net	0	0	1	1	0	0	0	0	0	1	1	1
1	ecnavi.jp	0	0	1	1	1	0	0	0	0	1	1	1
2	hubpages.com	0	0	1	1	0	0	0	0	0	1	0	1
3	extratorrent.cc	0	0	1	3	0	0	0	0	0	1	0	1
4	icicibank.com	0	0	1	3	0	0	0	0	0	1	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	wvk12-my.sharepoint.com	0	0	1	5	0	0	1	1	0	1	1	1
9996	adplife.com	0	0	1	4	0	0	0	0	0	1	0	1
9997	kurortnoye.com.ua	0	1	1	3	0	0	1	0	0	0	1	1
9998	norcaltc-my.sharepoint.com	0	0	1	5	0	0	1	1	0	1	1	1
9999	sieck-kuehlsysteme.de	0	1	1	4	0	0	1	1	0	1	1	1

10000 rows × 18 columns