Annals of Data Science (2024) 11(1):217–242
https://doi.org/10.1007/s40745-022-00379-8

Modeling Hybrid Feature-Based Phishing Websites Detection Using Machine Learning Techniques

Sumitra Das Guptta[1] · Khandaker Tayef Shahriar[1] · Hamed Alqahtani[2] · Dheyaaldin Alsalman[3] · Iqbal H. Sarker[1]

Received: 10 October 2021 / Revised: 4 January 2022 / Accepted: 8 January 2022 /
Published online: 21 March 2022

Abstract

In this paper, we mainly present a machine learning based approach to detect *real-time phishing* websites by taking into account *URL and hyperlink based hybrid features* to achieve high accuracy without relying on any third-party systems. In phishing, the attackers typically try to deceive internet users by masking a webpage as an official genuine webpage to steal sensitive information such as usernames, passwords, social security numbers, credit card information, etc. Anti-phishing solutions like blacklist or whitelist, heuristic, and visual similarity based methods cannot detect zero-hour phishing attacks or brand-new websites. Moreover, earlier approaches are complex and unsuitable for real-time environments due to the dependency on third-party sources, such as a search engine. Hence, detecting recently developed phishing websites in a real-time environment is a great *challenge* in the domain of cybersecurity. To overcome these problems, this paper proposes a *hybrid feature based anti-phishing* strategy that extracts features from URL and hyperlink information of client-side only. We also develop a new dataset for the purpose of conducting experiments using popular machine learning classification techniques. Our experimental result shows that the proposed phishing detection approach is more effective having higher detection accuracy of 99.17% with the XG Boost technique than traditional approaches.

Keywords Cybersecurity · Phishing detection · Machine learning · Hyperlink feature · URL feature · Anti-phishing · XG Boost · Hybrid feature

✉ Iqbal H. Sarker
iqbal@cuet.ac.bd; iqbal.sarker.cse@gmail.com

1 Department of Computer Science and Engineering, Chittagong University of Engineering & Technology, Chittagong 4349, Bangladesh

2 Unit of Cybersecurity, Department of Computer Science, Center of Artificial Intelligence, King Khalid University, Abha, Saudi Arabia

3 School of Engineering, Computing and Informatics, Dar Al-Hekma University, Jeddah, Saudi Arabia

1 Introduction

Phishing nowadays is one of the most serious and dangerous online threat in the domain of cybersecurity [1]. The use of social networks, e-commerce, electronic banking, and other online services has been increased immensely due to the rapid development of internet technologies. We Are Social (Global Overview Report 2021) [2] released "A Digital Report in 2021" data that the internet users have grown to 4.66 billion with an increase of 7.3 percent (316 million new users) compared to January 2020. At present, internet penetration stands at 59.5 percent which provides an opportunity to make money for a phishing attacker by blackmailing and stealing confidential information from internet users [3]. The attacker develops a fraudulent website and sends links to online platforms like Facebook, Twitter, emails, etc by conveying a message of panic, urgency, or a financial bid, and instructs the recipient to take immediate action [4]. When a user unwittingly clicks the link and updates any sensitive credentials, cyber attackers gain access to the user's information like financial data, personal information, username, password, etc. This stolen information is used by cybercriminals for a variety of illegal activities, including blackmailing victims. According to [5], there are five reasons why users fall for phishing:

1. Users do not have a deep understanding of URLs.
2. Users are unsure of which websites they should rely on.
3. Due to redirection or secret URLs, users are not able to see the entire address of the web page.
4. Users don't have much time to look up a URL or unconsciously visit certain web pages.
5. Users are unable to differentiate between legal and phishing websites.

Phishing attacks are now used to spread malicious software like ransomware [4]. Anti-Phishing Working Group (APWG) investigates phishing attacks and published a report showing that the number of phishing attacks identified by the APWG members increased by more than doubling in 2020 (APWG Q4 2020 Report 2020) [6]. 225,304 new phishing sites were discovered in October alone during the period of COVID-19 smashing all previous monthly records. In 2020, a record number of complaints (241,342) from the American public regarding phishing scams were received by Internet Crime Complaint Center(IC3) [7] with reported losses of more than $ $\tilde{5}$4 million. Hence, in this paper, we focus on detecting phishing websites effectively to help non-conscious internet users from being trapped by phishers and thus reduce the emotional and financial damages.

In the context of computing, "Data science" has become a hot area nowadays, as almost everything in our daily lives is digitally recorded as data and the extracted insights from the data is the key for providing intelligent solutions, discussed briefly in Sarker et al. [8]. Such data-driven solutions can be used for building an effective model as well as intelligent decision-making in various real-world application areas, such as business or financial analysis, cybersecurity, IoT applications, etc. [9–11]. Thus in this paper, we aim to build an effective data-driven solution using machine learning techniques to determine whether or not a website is phishing. Most of the machine learning based phishing detection approaches extract the features from the

URL, search engine, third-party, web traffic, DNS, etc. These types of approaches might not suitable for real-time phishing detection because of complexities and time constraints. According to the statistics of APWG, 1H2014 [12], phishing sites have a median life cycle of fewer than 10 hours and half of the phishing websites were taken down in less than a day. However, most phishing pages using compromised domains persist on the Internet for more than a day. Therefore, the research question we address in this paper is - "How can we develop an efficient and intelligent phishing detection model by taking into account the issues, mentioned above?"

To answer this research question, in this paper, we propose a hybrid feature based phishing detection approach that effectively identifies phishing websites and handles the problems mentioned above. To detect phishing websites we use URL based features or address bar features, and hyperlink based features. Hence, our feature extraction process is not dependent on any search engine or third-party services. We analyze and extract Hyperlink features from the source code of the webpage. We collect 15 different features from the URL and 10 different categories of features from the hyperlink information. We combine all of the features to get a hybrid feature set of 25 features to train our classification model. Our paper's key contributions are as follows:

- We first create a dataset by collecting phishing and legitimate website URLs from the open-source platforms.
- We propose an approach that dynamically extracts hybrid features and significantly utilizes them for phishing detection precisely.
- Our proposed machine learning based method detects zero-hour phishing attacks having high degree of accuracy.
- We have performed a wide range of experiments to show the effectiveness of our proposed approach compared to traditional methods.

The following is how the rest of the paper is structured: Sect. 2 discusses the related works mentioning a number of anti-phishing approaches based on machine learning. In Sect. 3, we present the methodology of our proposed strategy. The dataset creation and various features extraction for training the machine learning classifiers are also presented in this section. The implementation details, evaluation metrics, and experimental results are discussed in Sect. 4. Next we present the discussion in Sect. 5 and finally, Sect. 6 concludes this paper.

2 Related Work

Nowadays, many anti-phishing solutions are proposed by different authors for detecting phishing websites. Generally, we divide the phishing website detection approaches into 2 categories: User awareness/education based and software based. In this section, we illustrate a summary of these approaches as follows.

2.1 User Awareness/Education

The user education approach mainly depends on how efficiently educate the internet users so that they can understand the characteristics of phishing attacks. This under-

standing helps them to correctly identify phishing websites and emails. An interactive teaching game is developed by Sheng et al. [13] called "Anti-Phishing Phill", which teaches players how to recognize phishing websites. Users who played the game were better able to recognize phishing websites than those who did not. The main goal of this game is to give computer users conceptual knowledge about phishing assaults. Kumaraguru et. al. [14] designed an email-based anti-phishing education method that helps users to learn the cues to URLs to avoid falling into the trap of phishing scams. Their results suggested that user education provides a complementary way to help people better identify fraudulent emails and websites after establishing the first-line defense of automated phishing detection systems. Arachchilage et al. [15] developed a game design framework by deriving a theoretical model from Technology Thread Avoidance Theory (TTAT) to enhance user avoidance behaviour and protect them from phishing traps.

2.2 Software Based

Software based detection techniques are classified into listing based (blacklist/whitelist), visual-similarity based, machine learning based approaches.

(1) *List Based Techniques* List based detection approaches are two types: blacklist and whitelist techniques. Most of the browsers have their own list of blocked and safe Uniform Resource Locators (URLs). The database containing the blocked URLs is called a blacklist and the database of unblocked or safe URLs is called a whitelist. Wang et al. [16] and Han et al. [17] both utilize a whitelist based approach to classify URLs. Chiew et al. [18] worked with logo extraction and matching this logo using a whitelist. Other solutions for detecting suspicious URLs employ whitelists of resources such as layouts (Rosiello et al. [19]) and favicons (Chiew et al. [20]). In the blacklist method, a suspicious domain is checked whether it matches the blacklisted domains or not. If it is matched, then it is classified as phishing, otherwise legitimate. To detect new phishing URLs, Felegyhazi et al. [21] employed domain name and name server information from blacklisted URLs. The information from a URL's registration and DNS zone is compared to the information that is already stored in the blacklist.
The whitelist approach is exactly the opposite of the blacklist method. The main drawback of this list based technique is that it wrongly classifies the newly generated phishing websites whose age is less than a day, known as zero-day phishing attacks.

(2) *Visual Similarity based Techniques* In the visual similarity based techniques, the visual outlook of the suspicious website and its corresponding genuine website are compared. To compute the similarity across websites, these techniques [22] use a variety of features such as page source code, photos, textual content used in the website, HTML codes, CSS(Cascading Style Sheets), website logo, and so on. Most of the time, attackers copy the targeted website's outlook so that users can easily get tricked. So, in this technique, a similarity score is calculated for a suspicious site and the suspicious site is determined as phishing if its similarity

score is higher than a certain threshold. This technique is also not so effective for detecting zero-hour attacks.

(3) *Machine Learning based Approach* In the machine learning based approach, a dataset is created with extracted features. Then a classification algorithm is trained with the feature set and a website is classified as phishing if it matches with the predefined feature set. Machine learning approaches can detect even zero-day attacks if it is trained with heuristic features. Overall of the phishing website detection approaches, the machine learning approach is the better one. However, to gather and compute features from diverse sources such as address bar(URL), source code, website traffic, DNS, WHOIS database, search engine, etc., certain machine learning solutions necessitate large computations.

Our proposed approach is also based on machine learning. Here, some of the machine learning based approaches are discussed below.

Rao et al. [3] used both machine learning and image checking techniques to detect phishing websites by extracting heuristic features from URL, source code, and third-party services. Their model is able to detect zero-days attacks but has performance problems because of using third-party services. Jain et al. [4] proposed an anti-phishing approach based on machine learning algorithms using only hyperlink based features but has a problem of wrongly classifying the non-html websites. O. Sahingoz et al. [5] identified phishing websites by classifying them with NLP based features and word vectors but their system can not detect phishing websites that have shorter URLs. Y. Huang et al. [23] mainly used a capsule based neural network that consists of four branches. Their proposed model performed very well with a high accuracy rate, but the design architecture of the network is quite complex.

Rao et al. [24] developed a light-weight feature based application named CatchPhish that differentiate between phishing and legitimate website without visiting the website. They used two categories of features: hand-crafted features that are actually URL based and TF-IDF features. This application achieved 94.26% accuracy using a random forest classifier on their collected dataset. It can be used as first-level filtering of phishing websites within a shorter time period. Odeh et al. [25] achieved a very high accuracy rate of approximately 99% using the adaptive boosting approach. They have collected 30 features and these features are classified into four categories: Address bar based, abnormal based, HTML and JavaScript based, and domain based features. They took the most correlated features by utilizing features selection. Though they achieved a high accuracy rate, their approach is not fit for real-time phish detection.

Babagoli et al. [26] used a non-linear regression strategy for detecting a website whether it is phishing or not. They used a large dataset with 11055 web pages. Two types of meta-heuristic algorithms, Harmonic search (HS) and SVM are used for model training. This study concluded that HS gives better performance than SVM and they achieved 92.80% accuracy using the HS algorithm. R. Mohammad et al. [27] implemented a phishing attack detection model using the self-structuring neural network. The authors used the backpropagation algorithm for the weight adjustment of the network. They used 17 features collected from the URL, source code of the website, and also third-party services. The detection time is increased for using the third-party based features. However, their test set accuracy was 92.18% with 1000 epochs. F. Feng

et al. [28] designed a novel neural network for phishing detection. They designed their proposed novel neural network in such a way that the design risk is minimized and used the Monte Carlo (MC) algorithm for the training process. However, 97.71% accuracy was achieved by them with a low false-positive rate of 1.7%. They also showed that their proposed model performs better than other machine learning classifiers.

By summarizing the above works we can find that most of the works have the problems of using third-party services, low accuracy rate, and absence of identifying real-time phishing scams. Hence, an effective phishing detection approach would be helpful to detect real-time and zero-hours phishing attacks with high accuracy. Thus in this paper, we consider the development of an anti-phishing method by extracting the key features from the URLs and hyperlinks and using the hybridization of those features in the best possible machine learning model to detect real-time and diverse phishing sites more precisely.

3 Methodology

In this section, we present the system architecture of our proposed approach as shown in Fig. 1. We analyze and extract two categories of features to detect phishing attacks from suspicious web pages. Our features are based on the URL and hyperlink of the webpage. We extract URL based features by analyzing the structure of the URL and hyperlink based features from the source code of the website by generating a Document Object Model (DOM). It is easier to search any hyperlink-related information from the DOM tree because of its organized tree-like presentation of XML or HTML documents. At first, rules are generated for feature extraction. Generation of rules are very essential part in extracting hidden patterns and relationships in a dataset. So, URL structure and DOM tree are analysed in order to build reliable rules. Features are extracted based on these rules. All the features along with its creation rules are explained in detail in the upcoming subsection. The obtained features are combined together to create a hybrid feature set which is used to train the model for website classification using various machine learning classifiers. We find the best classifier by evaluating the performance with a lower error rate and higher accuracy. The process for detecting a phishing website is shown in Algorithm 1.

3.1 Dataset Development

We extract our desired features from 6000 different legitimate and phishing websites. We crawl Phishing websites when they are alive because of their limited lifespan. We collect phishing URLs from Phishtank [29] by developing a phish crawler to crawl the phishing URLs. We can quickly search all of the current valid phishing URLs in the 'Phish Search' tab on the Phishtank website. The 'ID' of the URL in the Phishtank database, the URL itself, the 'validity' property (search for 'valid phish'), and the 'online' properties are all presented on the search page in a table as shown in Fig. 2. We send a request to each phish search page by iterating through the page number and use the BeautifulSoup to access the source code of this page. We collect the phishing

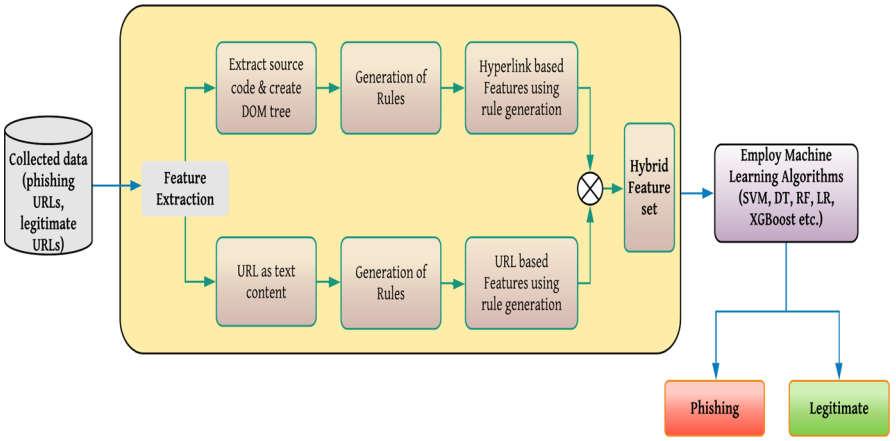


Fig. 1 Proposed methodology

Algorithm 1 Phishing Website Detection Model

Input: URL of suspicious website
Output: $Prediction \in \{0, 1\}$, 1 - phishing, 0 - legitimate
1: Procedure PhishDetection($inputURL$);
2: Rule generation for URL features
3: Extract URL based features (UF_1 - UF_{15})
4: Extract HTML source code & create DOM Tree
5: Rule generation from DOM tree for hyperlink features
6: Extract hyperlink based features(HF_1 - HF_{10}) from DOM Tree
7: Generate hybrid feature set by combining URL based and hyperlink based features
8: Remove unuseful feature UF_1
9: Apply hybrid feature set on well performed machine learning classifier(XGBoost)
10: **if** classifier predicts URL as phishing **then**
11: $Prediction \leftarrow 1$
12: **else**
13: $Prediction \leftarrow 0$
14: **end if**
15: **return** $Prediction$

IDs instead of URLs from each page's table because of the presence of '...' at the end of long URLs. Then we search these 'IDs' in Phishtank's 'phish detail' section that is shown in Fig. 3. We extract our desired valid phishing URL by using 'IDs' and sending requests to access the page source of the phish detail pages. In this way, we have crawled 3000 phishing URLs and collected the legitimate/benign URLs from the dataset provided by the University of New Brunswick [30]. This dataset contains over 35,300 legitimate URLs that were collected from the Alexa top websites [31]. We pick 3000 legitimate URLs randomly for our experiment from this dataset. Labeled values are set to 0 for legitimate URLs and 1 for phishing URLs.

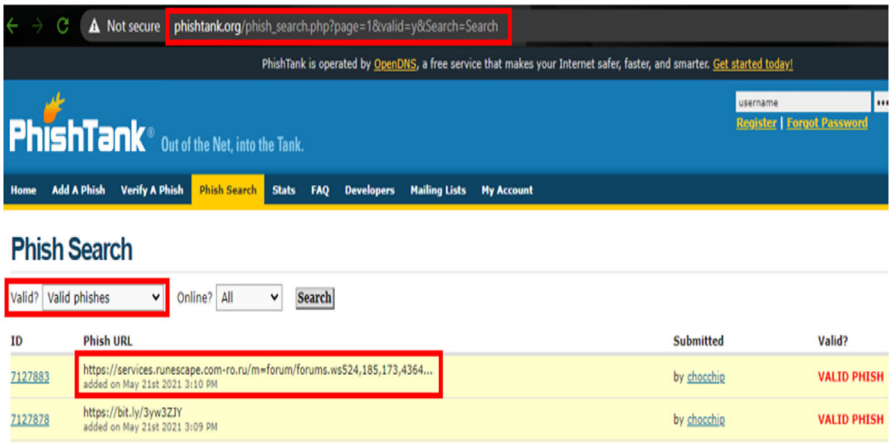


Fig. 2 A Phishtank's search page with valid phishing URLs searching

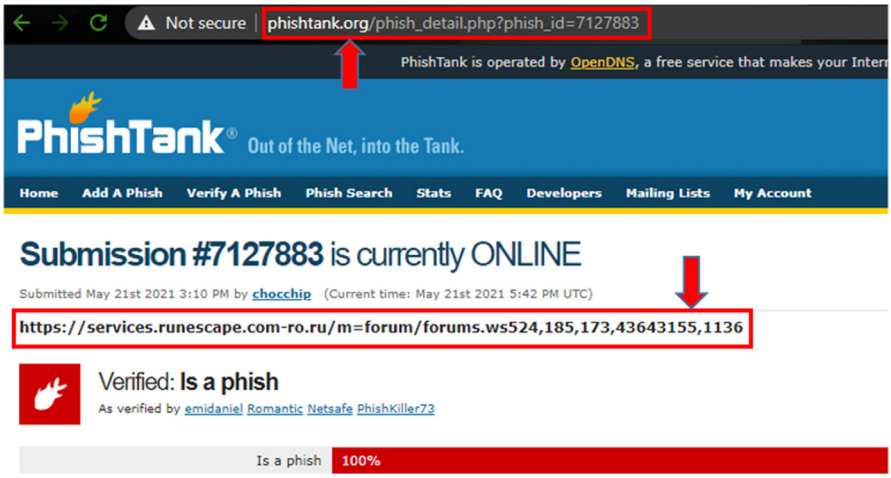


Fig. 3 A Phishtank's phish detail page for a phish URL

3.2 Feature Extraction

We develop a feature extractor that converts the labeled raw URLs to embedding feature representation as the training cannot be performed on strings. We have extracted 25 features that are divided into two categories (URL features and hyperlink features) as shown in Table 1 for the classification of web pages. We only use the client-side specific features because of the limitation of search engines and third-party dependent approaches. Some of the feature value is in the form of 0 or 1 indicating legitimate and phishing respectively. A phishing site that exactly looks like the login page of the official Paypal website is shown in Fig. 4. We highlight some of the unique phishing indication features also. The name of the features and their descriptions along with the criteria for assigning values to the features are described in the following.

Table 1 Categorical features used in our approach

Serial no.	Category	Features name	Total features
1	URL based features	Domain of URL, Count Subdomains in URL, IP Address in URL, "@" Symbol in URL, Length of URL,Depth of URL, Redirection "//" in URL, "http/https" in Domain name, HTTPS_scheme, Using URL Shortening Services "TinyURL", Prefix or Suffix "-" in Domain, Existence of sensitive word, Existence of trendy brand name, Existence of upper case letter, Number of dots in url	15
2	Hyperlink based features	No hyperlink, Internal Hyperlink ratio, External hyperlink ratio, Internal/external CSS, Suspicious Form link action, Null hyperlink, Internal/External Favicon, Common page detection ratio, Common page in footer section ratio, Server Form Handler	10

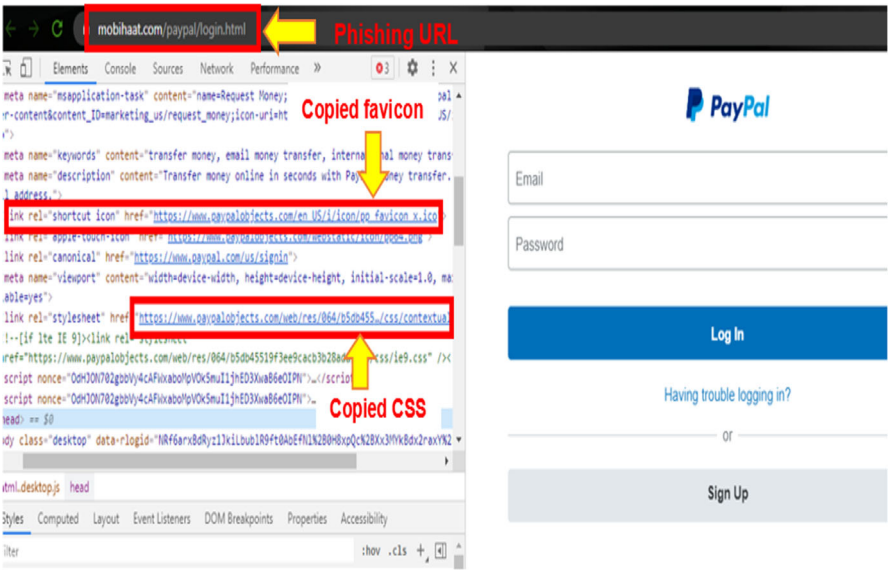


Fig. 4 A phishing website along with it's suspicious URL & source code

3.2.1 URL Based Feature

Uniform Resource Locator (URL) is used to find images, audio or video files, hypertext pages, etc. on the internet. The Fig. 5 represents the URL structure where the URL is divided into several sections. Web resources are accessed by URLs that start with a protocol like https, http, ftp, etc. The most secure protocol is HTTPS (Hypertext Transfer Protocol Secure). The second part of the URL is a hostname or sometimes an IP address that indicates the location of the server where the resource is located.

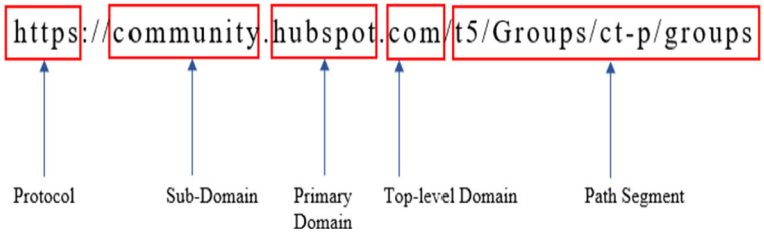


Fig. 5 Structure of URL(Uniform Resource Locator)

The hostname is divided into three sub-components. These are subdomain, primary domain, and top-level domain (TLD). The TLD is divided into several sections such as generic TLD (gTLD), country code TLD (ccTLD), etc. The third part of the URL is a path refers to a specific resource that the user has requested to access within the domain. The domain part and path are separated by a single '/' slash. The path structure has two optional fields. The first is a query that always starts with the question mark '?' And the other is a fragment preceded by the hash '#'. The standard URL format is as follows:
<Protocol>://<Sub domain>.<Primary domain>.<TLD>/<Path domain>
<?query><#fragment>

The phisher has complete control over the primary domain, subdomain, and path segment values. In this article, we explain how cybercriminal uses URL obfuscation to trick users.

1. *Domain name of URL* We store the entire domain name except "www." from the URL as a feature. This feature is not particularly useful for implementation. It will be removed from the feature set while training the model.

$$UF1 = \text{domain name of URL} \tag{1}$$

2. *Count subdomain in URL* This feature checks the number of dots in the hostname part of the URL. Typically, a legitimate URL has two dots in the domain name except 'www.'. Phishers add more subdomains and also add the domain name of the original website as a subdomain to deceive users by using multiple dots in the URLs. The URL is 'suspicious' when the number of dots in the hostname is equal to three and the feature value is set to 0.5. The URL is classified as 'phishing' when the number of dots is greater than three indicating that the URL has multiple subdomains and the feature value is set to 1.

$$UF2 = \begin{cases} 1, & \text{if dots in URL} > 3 \\ 0.5, & \text{if dots in URL} == 3 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

3. *IP Address in Domain* In some cases, attackers use the IP address as an alternative to the domain name in the domain part of the URL. Someone is actually trying to

steal personal information when the IP address of any version (IPV4 or IPV6) is located in the domain part of the URL.

$$UF3 = \begin{cases} 1, & \text{if IP present} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

4. *"@" symbol in URL* We analyze the entire URL to see if it has a special "@" symbol. Typically, attackers add "@" to the end of the actual website domain to make it look like a legitimate website domain to users. When a user clicks on this link, the browser ignores everything before the "@" sign and downloads the actual address that comes after it. Feature value is set to 1 for the presence of "@" in the URL, else set to 0.

$$UF4 = \begin{cases} 1, & \text{if '@' present} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

5. *Length of URL* Attackers create their own long phishing URLs to steal data in order to hide suspicious parts of the URL. The chances of a website becoming phishing increase with a large number of characters in the URL. Scientifically, there is no definite list of lengths that separate phishing URLs from the official URLs. In our experiment, we found that the length of official URL is 75. If the length is more than 75 but less than 100, then we consider the URL as suspicious and assign a feature value to 0.5.

$$UF5 = \begin{cases} 0, & \text{if URL length} < 75 \\ 0.5, & \text{if URL length} \geq 75 \text{ and URL length} < 100 \\ 1, & \text{otherwise} \end{cases} \tag{5}$$

6. *Depth of URL* This feature calculates the number of subpages in the URL of a website. Subpages are separated by a single slash symbol ("/") in the URL path section. The official website file directory is not that long. Attackers, on the other hand, keep their fake pages on servers at a very deep level. Therefore, the file path of their webpage is usually longer than the actual one. Feature value is numerical based on the URL.

$$UF6 = \text{total no. of subpages/subfolders in the URL path} \tag{6}$$

7. *Redirection "//" in URL* The "//" symbol, known as a double slash, is used to redirect the user to other websites. Phishers use this slash symbol in URL to redirect the user to a fake website. We find the last appeared place of "//" in the entire URL. This symbol appears in the 6th or 7th place (after http: or https:) in real cases. Therefore, if the location is more than 7, then we can say that "//" appears elsewhere in the URL other than following the rule. This feature is a binary feature with the number 1 (phishing) or 0 (legitimate).

$$UF7 = \begin{cases} 1, & \text{if ``//'' is found anywhere other than} \\ & \text{after the http/https protocol} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

8. *"http/https" in domain name* Phishers can add a "https" or "http" token to the domain part of a URL to confuse users. In this feature, we detect the presence of "http/https" in the domain part of a URL. If the domain part of the URL contains "http/https," the value assigned to this feature is 1 (phishing), otherwise 0 (legitimate).

$$UF8 = \begin{cases} 1, & \text{if http/https exists in domain} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

9. *https in scheme* This feature checks the protocol of the URL. If the protocol of the URL is "https", then the feature value is assigned to 0(legitimate) or otherwise 1(phishing). The protocol is a very important portion of a URL. Most of the legitimate websites use the "https" protocol for more secure connections while forwarding confidential information. But nowadays, phishers also use fake "https" connections to trick the users. So, this feature has not had so much effect in detecting phishing websites from a legitimate one.

$$UF9 = \begin{cases} 1, & \text{if URL protocol is https} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

10. *Using URL shortening service "tinyURL"* On the World Wide Web, shortening URL is a way to make Uniform Resource Locator (URLs) much shorter while linking to the desired webpage. This is achieved through the use of redirect, which points to a web page with a long URL. For example, the URL "https://w.wiki/U" is a shortened form of https://en.wikipedia.org/wiki/URL_shortening. But nowadays, many phishers are using URL shortening services to trick users. Therefore, in this feature, we check whether the URL uses URL shortening services or not.

$$UF10 = \begin{cases} 1, & \text{if tiny URL exists} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

11. *Prefix or Suffix "-" in domain* This feature checks for the presence of any "-" in the domain part of the URL. In legitimate domain names, dashes are rarely used. To deceive the victim, attackers often resort to adding prefixes or suffixes that are separated by "-" in the domain name. This address appears to be real for users, and they are trapped as they try to access this phishing website.

$$UF11 = \begin{cases} 1, & \text{if domain part includes “-”} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

12. *Existence of sensitive word* Most commonly used tokens or words in phishing URLs are ‘login’, ‘update’, ‘validate’, ‘activate’, ‘secure’ etc. Attackers use these words in a URL to convey urgency and encourage users to visit a phishing site to steal sensitive information quickly. We compile a list of 18 such terms as phishing terms. If the given URL contains any of the phishing terms in the list, the value of the feature is set to 1 (phishing), otherwise set to 0 (legitimate).

$$UF12 = \begin{cases} 1, & \text{if any sensitive word in URL} \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

13. *Existence of trendy brand name* Phishing websites are often generated by imitating well-known brand websites. As a result, hackers use trendy brand names in the URLs of phishing sites to deceive users. The user considers the phishing website to be the official website of the brand by seeing the brand name in the URL. We have compiled a list of the top 19 brands that phishers often target.

$$UF13 = \begin{cases} 1, & \text{if any trendy brand name exists in URL} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

14. *Existence of upper case letter* Legitimate URLs are usually written in lowercase letters. But phishing URLs often use uppercase letters to deceive users. If a URL is consisted by any uppercase letter then it is considered a phishing URL.

$$UF14 = \begin{cases} 1, & \text{if any uppercase letter in URL} \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

15. *Number of dots in URL* There are usually no more than two dots in the URL of legitimate website except the ‘www. ’. Therefore, in this feature, the number of dots in the entire URL is calculated. If it is more than 2, then this URL is classified as a phishing one, otherwise it is legitimate.

$$UF15 = \begin{cases} 1, & \text{if no.of dots} > 2 \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

3.2.2 Hyperlink Based Feature

In this section, we analyze the hyperlink features extracted from the source code of the website. We examine the Hyperlink information from the Document Object Model (DOM) tree representation which is a method of representing a webpage in an organized hierarchical manner so that users may navigate the document more easily.

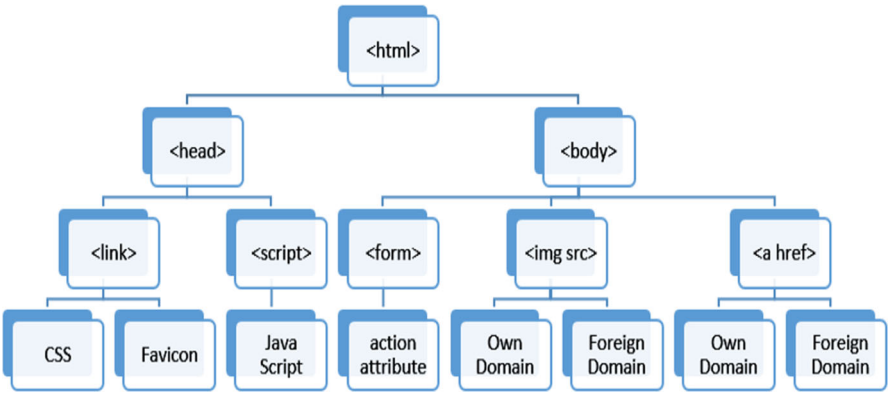


Fig. 6 HTML DOM Tree

DOM allows us to easily access and manipulate tags, IDs, classes, attributes, and elements. For this feature extraction, we mostly look at link, form, src, and anchor tags. In Fig. 6, a DOM tree is shown. We have extracted 10 hyperlink based features.

1. *No Hyperlink* A legitimate website has many web pages. On the other hand, phishing websites have a limited number of web pages. In addition, if attackers use a hidden hyperlink strategy, then the phishing website does not provide any hyperlinks [32]. If the website is authentic, we can find at least one hyperlink in the source code of this website [4]. The total number of hyperlinks is determined by considering href, link, and src tags. If the total hyperlink value is 0, then this feature value is set to 1 (phishing), the other is set to 0 (legitimate).

$$HF1 = \begin{cases} 0, & \text{total hyperlink} > 0 \\ 1, & \text{total hyperlink} = 0 \end{cases} \tag{16}$$

2. *Internal Hyperlink ratio* An internal website link means a link that points to the local/base domain. Hackers often create phishing websites to steal sensitive information that looks like real websites. Therefore, they copy the source code of the official website to easily generate phishing sites for stealing sensitive information. As they copy the source code from the original website, it may contain many links to the targeted website. Most hyperlinks on the legitimate website have the same base domain, while many hyperlinks on the phishing website have the domain of the corresponding legitimate website. For this feature, we calculate the ratio of internal hyperlinks with respect to the total links found in the source code. If the ratio is greater than or equal to 0.5, the website is considered valid and sets the

value of the feature to 0, otherwise set to 1.

$$\text{Ratio of internal link} = \begin{cases} \frac{\text{total internal hyperlink}}{\text{total hyperlink}}, & \text{if total hyperlink} > 0 \\ 0, & \text{total hyperlink} = 0 \end{cases} \tag{17}$$

$$HF2 = \begin{cases} 0, & \text{if Ratio of internal link} > 0.5 \\ 1, & \text{otherwise} \end{cases} \tag{18}$$

3. *External Hyperlink ratio* External hyperlinks refer to links with different domains or foreign domains. Most of the links of the website for phishing scams are external. In contrast, official websites use a few external hyperlinks. Therefore, we compute the ratio of external hyperlinks with respect to total hyperlinks available in the source code of the website. Since official websites contain only a few external links, the external hyperlink rate of the official website is usually small. If the ratio is less than 0.5, this feature value is set to 0, if not 1.

$$\text{Ratio of external link} = \begin{cases} \frac{\text{total external hyperlink}}{\text{total hyperlink}}, & \text{if total hyperlink} > 0 \\ 0, & \text{total hyperlink} = 0 \end{cases} \tag{19}$$

$$HF3 = \begin{cases} 1, & \text{if Ratio of external link} > 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

4. *Internal/External CSS* CSS (Cascading Style Styles) is a language that represents document formatting and influences the visual appearance of webpages written in HTML, XHTML, and XML. Phishing websites are generated by mimicking the design and visual appearance of genuine websites to attract potential victims. Building a phishing site for stealing sensitive information is often a low-level effort by attackers. Because of this, instead of creating their own CSS file, they try to use the CSS file of the official website they are targeting. CSS has two types: internal and external. External CSS file links are added using the <link> tag. So, we search for <link> tag that has at least two attributes as rel = 'stylesheet' and href = 'url' to find external links of a CSS file. Internal CSS is embedded within the HTML code of websites. Attackers mainly use external CSS files of the official websites to create phishing websites to steal sensitive information. Therefore, to set a feature value, we check if there are any external CSS files found in the source code of the website. If the external CSS file is a foreign hyperlink, then the feature value is set to 1, otherwise 0.

$$HF4 = \begin{cases} 1, & \text{if CSS file is external and} \\ & \text{current domain = base domain} \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

5. *Suspicious Form Link Action* Phishing websites often have a login or sign-in form for hijacking the personal and financial information of victims. When any user unknowingly fills out this form on a fake website, all the information entered on

the form is passed on to attackers. Usually, the action field of <form> tag contains the current website URL of a genuine website. But the action field of fake login form contains external links or a PHP file [4]. Sometimes the action field is 'null' or contains '#', 'javascript:void()', etc. Therefore, we check the value of the action field of the <form> tag to verify the login form's genuineness. This feature value is binary.

$$HF5 = \begin{cases} 1, & \text{if external link or php file or " ", "\#",} \\ & \text{"javascript:void(0)" in action field} \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

Null Hyperlink For this feature, we only check the anchor tag <a>. This feature calculates the percentage of Null links on a website compared to the number of anchor links. The attacker's plan is to keep the Internet users on the same page until they enter confidential information. As a result, anytime a user clicks on any of the links on the login page, they are taken back to the same login page. This is done by the following code.

$$\begin{gathered} < \text{a href} = \text{"\#"} > \\ < \text{a href} = \text{"\#content"} > \\ < \text{a href} = \text{"javascript:void(0)"} > \end{gathered}$$

The feature yields 1 if the null anchor link over total anchor link ratio is more than 0.34, otherwise, the result is 0.

$$\text{Ratio of null anchorlink} = \begin{cases} \frac{\text{total null anchor link}}{\text{total anchor link}}, & \text{if total anchor link} > 0 \\ 0, & \text{total anchor link} = 0 \end{cases} \tag{23}$$

$$HF6 = \begin{cases} 1, & \text{if Ratio of null anchor link} > 0.34 \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

6. *Internal/External Favicon* A favicon is a small icon that acts as a website identity. Favicon is added by using the <link> tag to the webpage. If the favicon displaying in the address bar belongs to an external domain, the website is considered as a phishing one for stealing sensitive information. Because the attacker often copies the favicon of the official website. Many users are deceived by a bogus address bar that shows a duplicate favicon. Therefore, we analyze the <link> tag containing the favicon and check the link for the same domain or not. If it is an internal favicon, then the feature value is 0 (legitimate), the rest is set to 1 (phishing).

$$HF7 = \begin{cases} 1, & \text{if external favicon} \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

7. *Common Page Detection ratio* Attackers spend less work and time setting up fake websites quickly. They populate the website with several anchor links to make it look legitimate. But they do not actually develop many web pages to include in

anchor links. Phishers may redirect a few or all links to a single common page. In phishing sites, this type of scenario leads to a high rate of common page detection. Therefore,this feature compares the frequency of the most common anchor link to the total number of anchor links.

$$HF8 = \begin{cases} \frac{\text{freq. of most common anchor link}}{\text{total anchor link}}, & \text{if total anchor link} > 0 \\ 0, & \text{total anchor link} = 0 \end{cases} \tag{26}$$

8. *Common Page in Footer section ratio* This feature is similar to the previous one, but highlights the more common page detection in the footer section.

$$HF9 = \begin{cases} \frac{\text{freq. of most common link in footer}}{\text{total anchor link in footer}}, & \text{if total anchor link} > 0 \\ 0, & \text{total anchor link} = 0 \end{cases} \tag{27}$$

9. *Server Form Handler(SFH)* If the SFH contains an empty or 'about: blank' string, it can be considered as a phishing attempt to steal sensitive information. Because steps must be taken based on user information provided in the form. In addition, if the SFH domain name is an external domain, this indicates that the website is suspicious. Therefore, we set the feature value to 0.5 on a suspicious webpage, 0 for legitimate, and set 1 for phishing one. This is a ternary feature.

$$HF10 = \begin{cases} 1, & \text{if SFH contains " " or "about:blank"} \\ 0.5, & \text{else if SFHs domain name is foreign domain} \\ 0, & \text{otherwise} \end{cases} \tag{28}$$

3.2.3 Hybrid Features

To increase accuracy, we have combined both feature categories (URL based and Hyperlink based) to obtain a hybrid feature set. In total, we have got 25 features. We remove the domain name feature (UF1) as a part of preprocessing step, before splitting the dataset into the train and test dataset. Because it is not a numerical feature and this feature is not relevant to distinguish a website from phishing.

3.3 Employ Machine Learning Algorithms

We split the dataset into train and test sets. 80% of the dataset is considered as a train set and 20% of the remaining is considered as a test set. The labeled train set is used for training the classifiers and the test set is used to evaluate the performance of each classifier and to obtain the best classifier. Five popular ML techniques are used to do classification work such as Random Forest, Decision tree, Support Vector Machine, Logistic Regression, XG Boost [33]. The reason for choosing these classification techniques is to their better learning capabilities from cyber data [34].

Logistic Regression is a popular technique for assessing the chances that an instance belongs to a specific class (e.g., How likely is it that this URL is a phishing attempt?). If

Table 2 Optimized parameters for ML models

Classifier	Parameters
Logistic regression	solver='lbfgs', C=1.0, max_iter = 100, penalty = 'l2'
Decision tree	criterion='gini', max_depth=5
Support vector Machine	kernel='linear', C=10, random_state=12
Random forest	n_estimators=10, random_state=42
XG Boost	learning_rate=0.4, max_depth=5

the estimated probability is more than 50%, the model predicts that the instance belongs to that class (phishing, labeled "1"), otherwise it predicts that it does not (legitimate, labeled "0"). That is how it functions as a binary classifier. Another supervised learning model that can be used for binary data classification is the Support Vector Machine. Generally, it is used to draw lines between classes. The margins are drawn so that the space between the margin and the classes is as maximum by reducing the classification error [35]. Furthermore, SVMs have been successfully employed in the past to solve classification problems such as phishing website classification and review spammer detection, etc [36]. By recursive division of data, the decision tree algorithm produces a classification tree for the provided dataset. Each node in the decision tree represents a feature, with each stem presenting a feature value and a possibility, and the final node presenting the outcome [37]. In random forest algorithm, a forest is constructed with several decision trees. This classifier becomes popular because of its faster execution speed [38].

We select the best ML model based on the accuracy achieved on test data. In this case, the XG Boost classifier performs better compared to other ML algorithms as explained in the experiments section. Hence, we implement the XG Boost classifier for the final prediction of phishing websites. Extreme gradient boosting (XGBoost) is a boosting method that is based on the ensemble approach. Distributed machine learning community (DMLC) owns the XGBoost and it works so well because in the dataset every bit of data value is checked by it [39]. A summary of the selected parameters for the ML models mentioned above is provided in Table 2.

4 Evaluation and Experimental Results

This section describes the implementation details and performance. We have performed a range of experiments on the datasets that we have created to measure and validate the effectiveness of our proposed machine learning based approach. The detailed comparative analysis among various methods with existing techniques is also illustrated in this section.

4.1 Implementation Tools

We use a laptop having a Core i3 processor with 2.0 GHz clock speed and 4 GB RAM to implement our proposed approach to detect phishing websites. We use Python

Programming Language due to its extensive support for libraries and short compile time. We create different functions to extract necessary features. Some of the libraries used in the extraction process are: **re:** We use this library to find the desired string from the URL; **urllib2:** We use this library to get the response object from any URL and parse the URL components; **BeautifulSoup:** We use this useful library to extract information from XML and HTML documents and to create DOM (Document Object Model); **Favicon:** Website's favicon address is extracted by using this python library.

4.2 Evaluation Metrics

To evaluate the effectiveness of the proposed approach, we use the true positive rate, true negative rate, false positive rate, false negative rate, accuracy, f1 score, precision and AUC (Area Under the Curve).

True Positive Rate (TPR) It computes the ratio of correctly identified phishing websites with respect to the total phishing websites and it is also known as Recall.

False Positive Rate (FPR) It computes the ratio of genuine websites wrongly classified as phishing with respect to the entire legitimate websites.

Accuracy The overall rate of accurate predictions is determined by accuracy.

Precision It computes the ratio of accurately identified phishing websites with respect to all the websites identified as phishing.

F1 Score F1 score refers to the harmonic mean of Precision and Recall.

False negative rate (FNR) FNR refers to the percentage of phishing websites that are incorrectly identified as genuine.

ROC Curve The illustration of the diagnostic ability of detecting phishing website of binary classifier system can be plotted graphically by using a Receiver Operating Characteristics curve, or ROC curve.

Area Under the Curve (AUC) The entire two-dimensional area underneath the entire ROC curve is measured by AUC and it delivers an aggregate measure of performance for all possible thresholds of classification.

The equation of precision, recall, accuracy, F1 Score, FPR are given as below:

$$Recall = TPR = \frac{TP}{TP + FN} \tag{29}$$

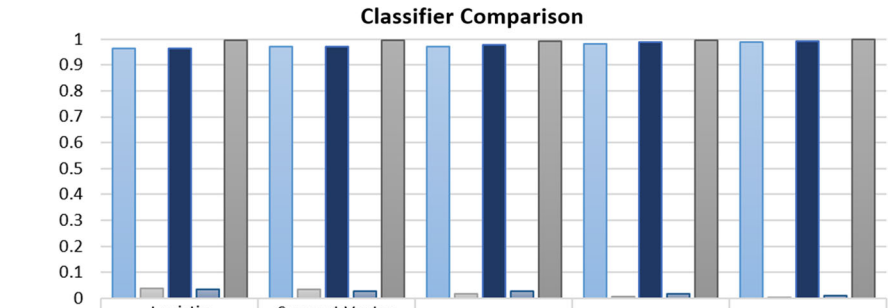
$$Precision = \frac{TP}{TP + FP} \tag{30}$$

$$FPR = \frac{FP}{TN + FP} \tag{31}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{32}$$

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{33}$$

Here, true positive, true negative, false positive and false negative are indicated by TP, TN, FP and FN respectively.



	Logistic Regression	Support Vector Machine	Decision Tree	Random Forest	XG Boost
TPR	0.9642	0.9727	0.971	0.983	0.9881
FPR	0.0392	0.0343	0.0179	0.0082	0.0049
Accuracy	0.9625	0.9692	0.9767	0.9875	0.9917
FNR	0.0358	0.0273	0.029	0.017	0.0119
AUC	0.9955	0.9948	0.9931	0.997	0.9989

Fig. 7 Performance comparison of our approach using various classifiers

4.3 Results on Popular Machine Learning Classifiers

We implement 5 different classifiers (Random Forest, Decision tree, XG Boost, Support Vector Machine, Logistic Regression) and then compare their performance using true positive rate, false positive rate, accuracy and false negative rate. Performance comparisons are shown in the Fig. 7. We also look at the area under the ROC (Receiver Operating Characteristic) curve to find the most accurate metrics. From the comparison chart in 7, we analyze that the XG Boost classifier performs better with the high accuracy and TPR and the low FPR and FNR. In our experiment, the area under the ROC curve of finding a phishing website generated by the XG Boost classifier is 99.89%. In Fig. 8, a ROC curve with the Area value is displayed for each classifier.

4.4 Evaluation of Features

In this experiment, We evaluate the performance of each individual category of feature set and also measure the performance of the hybrid feature set by using the XG Boost classifier to categorize the website. Figure 9 shows the comparison of performance metrics for each and every category of feature set. URL-based features work well with a 98.42% accuracy rate and 97.79% true positive rate, as shown in the figure. Using only hyperlink-based features produces 84.67% accuracy and 96.93% true positive rate. We present the results of our proposed approach by combining all features (UF2-UF15 and HF1-HF10) to obtain the hybrid feature set that produces a high true positive rate (approximately 99%) with a low false positive rate (less than 0.5%). Both URL-based and hyperlink-based features are useful for individual web-based detection, but they are not sufficient to detect various types of phishing URLs. If a hybrid feature

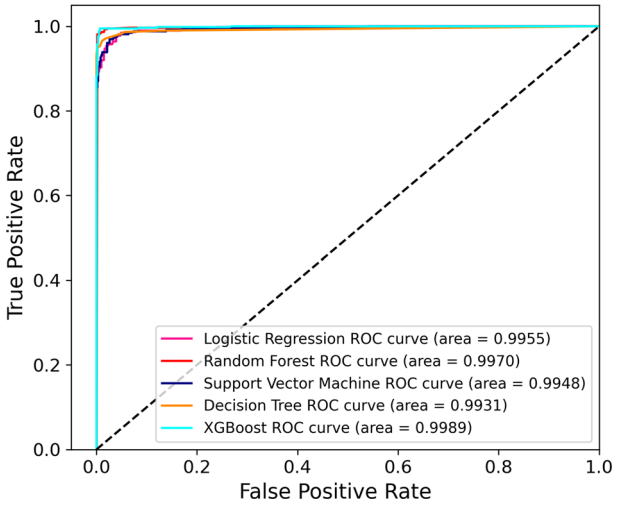


Fig. 8 Comparison of classifiers with ROC curve

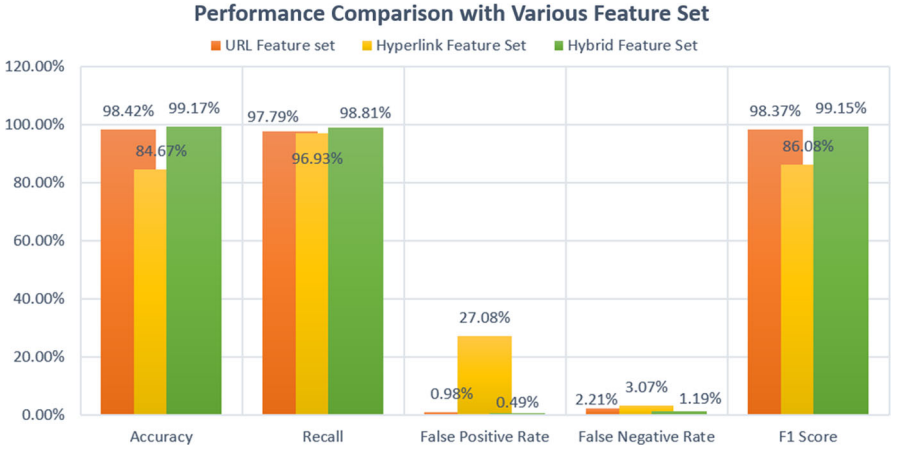


Fig. 9 Performance measures based on Various Feature Sets

set is used to detect phishing URLs, we could be able to detect phishing attacks more accurately.

4.5 Comparison with Other Approaches

We compare the proposed approach with other existing anti-phishing approaches for measuring the effectiveness, as shown in Table 3. We assess the comparison based on

the characteristics of accuracy rate, true positive rate, language-independent, search engine independent, third-party independent, and zero-hour attack detection. The work of Rao et al. [3] offers a high degree of accuracy compared to our work, but their performance is highly dependent on search engines and third-party services. Only Rao et al. [3] and O. Sahingoz et al. [5] generates a higher TPR but lower performance than ours. Some applications (Odeh et al. [25], Babagoli et al. [26]) are unable to detect zero-hour phishing attacks, which is a very important issue nowadays. Most of these methods are not independent of search engines, but search engine results show the most popular sites on top of it and feature extraction is also time-consuming and complex. Therefore, we do not consider search engine based features in our proposed approach which helps us to improve the performance of the model.

5 Discussion

In this paper, we propose a data-driven approach for the purpose of detecting phishing websites using various machine learning classifiers, such as Decision Tree, XGBoost, Random Forest, Support Vector Machine, and Naive Bayes by implementing various numbers/types of features such as URL-based features, hyperlink-based features, and hybrid features. Developing an effective feature set is a challenging task. Therefore, we extract features from two categories to obtain the hybrid feature set. These categories are URL-based features and HTML source code-dependent hyperlink-based features. The experimental results show that the introduction of hybrid features, including URL and hyperlink features, improves the performance of the phishing detection system with a high accuracy rate of 99.17% compare to individual URL-based and hyperlink-based features. We develop our dataset with 6000 URLs containing 3000 legitimate URLs and 3000 phishing URLs for experimental purposes. We also have constructed a phish crawler for scraping the phishing URLs from PhishTank [29] website. This hybrid feature based machine learning model can be used to detect not only phishing threats but also other types of cyber security attacks such as anomaly detection, fraud detection, and zero-day attacks [1].

To improve the efficiency of the system, more cutting-edge technologies, such as advanced data analytics [8], deep learning mechanisms [41] or others AI techniques [42] can be implemented. Although, deep learning technologies require a large dataset, we have a plan to analyze with various types of deep learning techniques according to the data characteristics, summarized in Sarker et al. with a taxonomy [41]. Our proposed method may mistakenly find phishing websites extract user's information, if an attacker uses embedded objects such as iFrame, Images, Flash, etc instead of DOM to encrypt HTML coding. Thus our future goal is to design a system that can detect non-HTML websites with high accuracy by implementing additional features except third-party dependent features to improve the efficiency of our system.

Table 3 Comparison of different anti-phishing approaches based on performance

Approach	Accuracy	TPR	Language independent	Search-engine independent	Third party-independent	Zero hour attack detection
Rao et al. [3]	99.55	99.44	Yes	No	No	Yes
Jain et al. [4]	98.42	98.39	Yes	Yes	Yes	Yes
Sahingoz et al. [5]	97.98	99.0	Yes	Yes	Yes	Yes
Rao et al. [24]	94.26	93.31	Yes	Yes	Yes	Yes
Odeh et al. [25]	98.9	98.6	Yes	No	No	No
Zhang et al. [40]	95.83	96.2	No	Yes	No	Yes
Babagoli et al. [26]	92.80	96.3	Yes	No	No	No
Our proposed approach	99.17	98.81	Yes	Yes	Yes	Yes

6 Conclusion

Our proposed phishing detection approach is completely a client-side solution where hybridization of URL-based and Hyperlink-based features are implemented. We have developed a dataset by collecting URLs from different sources and including various websites there to validate our proposed approach. Experimental results on our dataset show that the proposed method is very effective having a 98.81% true positive rate and only 0.49% false positive rate. The proposed method depends entirely on the address bar and the source code of the website. The effective integration of both feature sets to create a hybrid feature set is one of the significant accomplishments of the paper. We believe that if some more particular features can be included, the accuracy of our approach will be further enhanced. However, third-party-dependent features will increase the complexity of the proposed method. Currently, mobile devices are ubiquitous, and they seem to be an ideal target of cyberattacks such as mobile phishing to steal sensitive mobile data. Therefore, mobile phishing will become a major threat in the future for us.

Acknowledgements We thank the reviewers for their insightful comments to improve the manuscript

Author Contributions All authors equally contributed to preparing and revising the manuscript.

Funding The authors are thankful to the Deanship of Scientific Research at King Khalid University for supporting this project under grant code RGP.2/61/43.

Declarations

Conflict of Interest The authors declare no conflict of interest.

Ethical statements The authors follow all the relevant ethical rules.

Data and code availability Data and codes used in this work can be made available upon reasonable request.

References

1. Sarker IH, Furhad MH, Nowrozy R (2021) Ai-driven cybersecurity: an overview, security intelligence modeling and research directions. SN Comput Sci 2(3):1–18
2. Digital 2021: The latest insights into the 'state of digital'. https://wearesocial.com/blog/2021/01/digital-2021-the-latest-insights-into-the-state-of-digital. Accessed 5 July 2021
3. Rao RS, Pais AR (2019) Detection of phishing websites using an efficient feature-based machine learning framework. Neural Comput Appl 31(8):3851–3873
4. Jain AK, Gupta BB (2019) A machine learning based approach for phishing detection using hyperlinks information. J Ambient Intell Human Comput 10(5):2015–2028
5. Şahingöz ÖK, Buber E, Demir Ö, Diri B (2017) Machine learning based phishing detection from uris
6. Apwg q4 2020 report. https://docs.apwg.org/reports/apwg_trends_report_q4_2020.pdf. Accessed 2 Jan 2021
7. Internet crime complaint center. https://www.ic3.gov/Media/PDF/AnnualReport/2020_IC3Report.pdf. Accessed 15 Aug 2021
8. Sarker Iqbal H (2021) Data science and analytics: an overview from data-driven smart computing, decision-making and applications perspective. SN Comput Sci
9. Shi Y, Tian Y, Kou G, Peng Y, Li J (eds) (2011) Optimization based data mining: theory and applications. Springer

10. Iqbal H, Sarker AC, Han J, Watters P (2022) Automated rule-based services with intelligent decision-making. Context-aware machine learning and mobile data analytics. Springer
11. Olson DL, Shi Y, Shi Y (2007) Introduction to business data mining, vol 10. McGraw-Hill/Irwin New York
12. Apwg h1 2014 report. http://docs.apwg.org/reports/APWG_GlobalPhishingSurvey_1H2014.pdf. Accessed 2 Oct 2020
13. Sheng S, Magnien B, Kumaraguru P, Acquisti A, Cranor LF, Hong J, Nunge E (2007) Anti-phishing phil: the design and evaluation of a game that teaches people not to fall for phish. In: Proceedings of the 3rd symposium on Usable privacy and security, pp 88–99
14. Kumaraguru P, Sheng S, Acquisti A, Cranor LF, Hong J (2010) Teaching johnny not to fall for phish. ACM Trans Internet Technol (TOIT) 10(2):1–31
15. Arachchilage NAG, Love S (2013) A game design framework for avoiding phishing attacks. Comput Hum Behav 29(3):706–714
16. Wang Y, Agrawal R, Choi B-Y (2008) Light weight anti-phishing with user whitelisting in a web browser. In: 2008 IEEE Region 5 Conference. IEEE, pp 1–4
17. Han W, Cao Y, Bertino E, Yong J (2012) Using automated individual white-list to protect web digital identities. Expert Syst Appl 39(15):11861–11869
18. Chiew KL, Chang FH, Tiong WK et al (2015) Utilisation of website logo for phishing detection. Comput Secur 54:16–26
19. Rosiello APE, Kirda E, Ferrandi F et al. (2007) A layout-similarity-based approach for detecting phishing pages. In: *2007 third international conference on security and privacy in communications networks and the workshops-securecomm 2007*. IEEE, pp 454–463
20. Chiew KL, Choo JS-F, Sze SN, Yong KSC (2018) Leverage website favicon to detect phishing websites. Secur Commun Netw
21. Felegyhazi M, Kreibich C, Paxson V (2010) On the potential of proactive domain blacklisting. LEET 10:6–6
22. Jain AK, Gupta BB (2017) Phishing detection: analysis of visual similarity based approaches. Secur Commun Netw
23. Huang Y, Qin J, Wen W (2019) Phishing url detection via capsule-based neural network. In: 2019 IEEE 13th international conference on anti-counterfeiting, security, and identification (ASID). IEEE, pp 22–26
24. Rao RS, Vaishnavi T, Pais AR (2020) Catchphish: detection of phishing websites by inspecting urls. J Ambient Intell Hum Comput 11(2):813–825
25. Odeh A, Keshta I, Abdelfattah E (2021) Phiboost-a novel phishing detection model using adaptive boosting approach. Jordanian J Comput Inf Technol (JJCIT) 7(01)
26. Babagoli M, Aghababa MP, Solouk V (2019) Heuristic nonlinear regression strategy for detecting phishing websites. Soft Comput 23(12):4315–4327
27. Mohammad RM, Thabtah F, McCluskey L (2014) Predicting phishing websites based on self-structuring neural network. Neural Comput Appl 25(2):443–458
28. Feng F, Zhou Q, Shen Z, Yang X, Han L, Wang J (2018) The application of a novel neural network in the detection of phishing websites. J Ambient Intell Hum Comput:1–15
29. Phishtank opensourcd platform. http://phishtank.org/. Accessed 2 Oct 2020
30. University of new brunswick,url dataset. https://www.unb.ca/cic/datasets/url-2016.html. Accessed 2 Oct 2020
31. Alexa - find top websites. https://www.alexa.com/. Accessed 2 Oct 2020
32. Geng G-G, Xiu-Tao Y, Wei W, Chi-Jie M (2014) A taxonomy of hyperlink hiding techniques. Asia-Pacific web conference. Springer, pp 165–176
33. Sarker IH (2021) Machine learning: algorithms, real-world applications and research directions. SN Comput Sci 2(3):1–21
34. Sarker IH (2021) Cyberlearning: effectiveness analysis of machine learning security modeling to detect cyber-anomalies and multi-attacks. Internet of Things 14:100393
35. Mahesh B (2019) Machine learning algorithms-a review. 01
36. Anupam S, Kar AK (2021) Phishing website detection using support vector machines and nature-inspired optimization algorithms. Telecommun Syst 76(1):17–32
37. Tang L, Mahmoud QH (2021) A survey of machine learning-based solutions for phishing website detection. Mach Learn Knowl Extract 3(3):672–694

38. Attack and anomaly detection in iot sensors in iot sites using machine learning approaches. Internet of Things 7:100059 (2019)
39. Bhati BS, Chugh G, Al-Turjman F, Bhati NS (2021) An improved ensemble based intrusion detection technique using xgboost. Trans Emerg Telecommun Technol 32(6):e4076
40. Zhang D, Yan Z, Jiang H, Kim T (2014) A domain-feature enhanced classification model for the detection of chinese phishing e-business websites. Inf Manag 51(7):845–853
41. Sarker IH (2021) Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput Sci 2(6):1–20
42. Sarker IH (2022) Ai-based modeling: techniques, applications and research issues towards automation, intelligent and smart systems. SN Comput Sci