Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 8241104, 24 pages
https://doi.org/10.1155/2021/8241104

Research Article

Hybrid Rule-Based Solution for Phishing URL Detection Using Convolutional Neural Network

Youness Mourtaji[1], **Mohammed Bouhorma**[1], **Daniyal Alghazzawi**[2], **Ghadah Aldabbagh**[3] **and Abdullah Alghamdi**[2]

[1]*Computer Science, Systems and Telecommunication Laboratory, Faculty of Sciences and Techniques, Abdelmalek Essaâdi University, Tangier, Morocco*
[2]*Information Systems Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia*
[3]*Computer Science Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia*

Correspondence should be addressed to Daniyal Alghazzawi; dghazzawi@kau.edu.sa

Received 28 April 2021; Revised 8 June 2021; Accepted 19 August 2021; Published 16 September 2021

Academic Editor: Chi-Hua Chen

The phenomenon of phishing has now been a common threat, since many individuals and webpages have been observed to be attacked by phishers. The common purpose of phishing activities is to obtain user's personal information for illegitimate usage. Considering the growing intensity of the issue, this study is aimed at developing a new hybrid rule-based solution by incorporating six different algorithm models that may efficiently detect and control the phishing issue. The study incorporates 37 features extracted from six different methods including the black listed method, lexical and host method, content method, identity method, identity similarity method, visual similarity method, and behavioral method. Furthermore, comparative analysis was undertaken between different machine learning and deep learning models which includes CART (decision trees), SVM (support vector machines), or KNN (K-nearest neighbors) and deep learning models such as MLP (multilayer perceptron) and CNN (convolutional neural networks). Findings of the study indicated that the method was effective in analysing the URL stress through different viewpoints, leading towards the validity of the model. However, the highest accuracy level was obtained for deep learning with the given values of 97.945 for the CNN model and 93.216 for the MLP model, respectively. The study therefore concludes that the new hybrid solution must be implemented at a practical level to reduce phishing activities, due to its high efficiency and accuracy.

1. Introduction

The term phishing comes from the word fishing with the way that hackers "lure" victims using a "bait" and "fishes" for any sensitive personal information [1–4]. Lastdrager [5] defined phishing as "a scalable act of deception whereby impersonation is used to obtain information from a target." To do so, the hacker uses different approaches either to beguile targets by a trick or to convey payload through indirect ways in order to get any confidential data from the unfortunate casualty or even compromise target's system [6]. Frequent or spear-phishing attacks could have severe results for all kinds of victims (simple users and private or public organizations and states), such as the loss of all types of personal and confidential information, financial loss and the compromise of national security [7], and losing trust [8, 9].

Phishing attempts to nefarious aims such as getting sensitive information by sending bulk emails pretending to be from an outstanding organization and reputable institution. Phishers use social engineering and spear-phishing techniques to deploy malware into a given network that steals

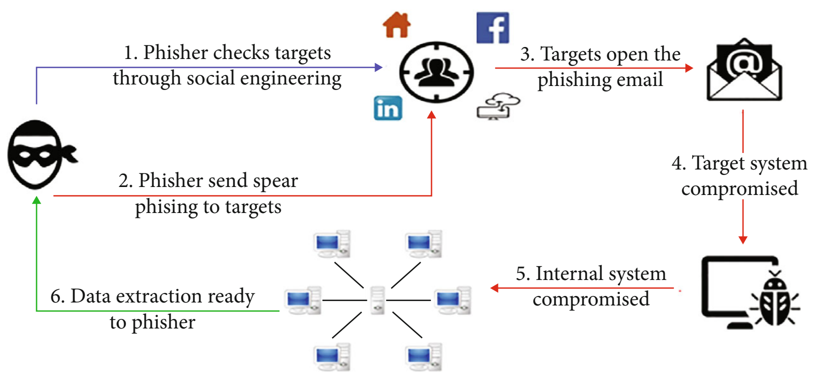
1. Phisher checks targets through social engineering
2. Phisher send spear phising to targets
3. Targets open the phishing email
4. Target system compromised
5. Internal system compromised
6. Data extraction ready to phisher

Figure 1: Phishing general process.

all-important operation data of the organization (Figure 1). Generally, this phenomenon has shown four principal forms of acting:

(i) Email-to-email: when a hacker sends an email asking for personal data from the victim in question

(ii) Email-to-website: when a hacker sends an email embedded with a phishing web address to the victim

(iii) Website-to-website: when an unfortunate victim taps on the phishing website through another website or an online advert

(iv) Browser-to-website: when somebody incorrectly spells an authentic web address and after that gets alluded to a phishing site that has a semantic similarity to the real web address.

The most used attack is to attract users (using social engineering techniques) into perfectly designed phishing websites that look like the original target organizations websites to get personal information from users by filling some forms. Phishing, including spear-phishing, has become a severe problem due to its unpredictability. This in turn let researchers and practitioners strive to find solutions to defend it or at least to make users aware of the danger of this phenomenon.

In the first quarter of 2015, CYREN reports 51 percent increase in phishing attacks [10]. Additionally, APWG [11] announced an expansion of 5753% of regular phishing assaults every month over a 14-year period, from 2004 to 2018. In 2015, the most significant part of a billion individual records was stolen, an expansion contrasted with the earlier year [12]. The Kaspersky Lab reported that phishing in the financial sector reached an all-time height in 2016 [13]. Also, the FBI has calculated a total loss of $2.3 billion by scamming business emails the period from October 2013 to February 2016 in USA [14]. Finally, in 2017, the China Federation of Anti-Phishing website reported that it encountered 391747 phishing sites and endured a global loss of around $111 billion from July 2011 to July 2012 [15].

Many techniques, based on different perspectives, were used to face these kinds of attacks, like blacklisted, lexical, content, identity, visual similarity, or behavioral features extraction-based methods. The list of methods is not exhaustive of course; rather, it helps to extract some useful features about the web page or the URL in question. These listed methods use either static/dynamic analysis or both of them (also known as hybrid), which is the case of solution provided in this study. The static analysis method features extraction which is performed without executing (loading) the webpage of URL in the test. After that, a numerical vector is constructed to compare it with a mathematical prebuilt model. Then, it is injected into a machine learning or deep learning classifier to conclude if this URL represents a threat of phishing or if it is a legitimate one. Alternatively, with regard to the dynamic analysis, the webpage is loaded at first before predicting the result. This is helpful because some source codes of webpages can change after the loading process, for example, direct download of some files (drive-by-download) or generating some additional malicious source code after clicking to see a video for example.

The phishing phenomenon is a very complex issue to understand and to analyse since it is a combination of technical and social ways in which there is no known magical formula to prevent it from happening or solve it all together. This gives rise to a need to create smart and practical solutions that should be maintained over time. Many different methodology-based solutions have been proposed; but they face one major problem which gives rise to a large number of false positives (safe web pages that are classified as phishing ones), mainly due to the limitation of such approaches, for example, depending only on fixed solution such as the black and white listing method, lack of human intelligence and experts, and lack of timeliness and scalability. Our ongoing research is aimed at creating a solution that can detect and prevent advanced and persistent threats using intelligent techniques for conducting logical rules and deep learning models that help to update the behavior of the solution using only the URL of the web page in question.

This study provides an improvised solution based on different kinds of features extraction and a comparative study considering the use of machine learning and deep neural network classifier-based models to detect or prevent phishing web pages. The overall process would include the construct of mathematical vector which will be compared to a prebuilt base mathematical model generally using a classifier such as machine learning or deep learning models to predict a particular value from a list of expected values. Its overall understanding is initiated by utilizing the URL of the webpage in the test as the main entrance to our system and classifying if this URL represents a phishing web page or a safe one. Each one of those features are categorized by the inclusion of three possible values {-1 "for safe," 0 "for suspicious," and 1 "for phishing"} with the aim of constructing a vector of discrete values [{-1, 0, 1}…{-1, 0, 1}], corresponding to [$f1, f2, f3, \cdots, f37$]. This vector will be injected and tested with the model built and get a final result as "1" which represents that the URL include phishing or as "-1" which represents the safety of URL as phishing.

To our knowledge, this is the first paper that attempts to compare the efficiency of the machine and deep learning models in binary classification. The study is important in providing the knowledge which serves as a greatest support for different organizations in escaping or dealing with the problem of phishing. This may lead in the development of strong customer base for organizations such as banks. Most of the organizations lose their valuable data due to the above discussed problem. In such circumstances, the development of the innovative solution as provided in this study would make an important contribution to the organizational industry. In addition, findings of this study will serve as an important contribution in the field of information technology, as highly secured measures for the individual data and information may support its users in purchasing and using more IT-based products. Finally, individuals, i.e., common people and academics, may significantly benefit from the study, as it provides detailed knowledge regarding different methods that may assist users in dealing with the issue.

2. Literature Review

Researches have shown many solutions to extract useful features used in phishing detection solutions; the review of previous studies includes the blacklist, lexical, content, visual, identity, and behavioral-based methods. The blacklist based method is a common technique that works with a principal to check if a domain name of the URL was already blacklisted in some databases by trusted companies like Google Safe Browsing [16], or some antiviruses. If so, the user will be warned that this web page is malicious since it is black listed, leading towards the visibility of unsafe content. Generally, this technique gives a good accuracy as companies spend long hours and resources to analyse the domain and the website directly either by themselves or by a user's request.

However, if the domain is in the blacklist database that does not necessarily mean that the current web page is malicious. One of the fundamental disadvantages of the given technique is its inability to detect or prevent spear-phishing (which means a web page that is created for a very fast purpose and then disappears). This requires permanent user's requests and user end experience to analyse web pages. The technique further depends on the browser or the tool used to access the web page. Therefore, if this tool is hacked or not updated, the technique will not be trusted. In any case, this feature will be used in our system as a first step, but it will not be considered a trusted feature.

Another method, i.e., the lexical method works to analyse the lexicography and to find patterns from a URL, such as its length, the presence of special characters like "@," "-," "//," or "#," for example, "http://www.example1.com@http://www.example2.net" where the user will be automatically redirected to "http://www.example2.net." The objective is to hide malicious techniques of redirection like using "@" or "//" symbols or some abnormal queries like SQL injection in long URLs [6]. Another example is the use of IP address instead of the normal domain, in which attackers can change permanently while eliminating the DNS translator. Generally, the URL's string is split into parts to extract the binary feature for each token in the hostname (delimited by ".") and in the path of the URL [17]. There is another kind of features extraction, which is called the host-based method. It consists of getting information like WHOIS, AS, and MX numbers. Features belonging to the host-based methods can be used with lexical features, as they can answer questions like "where" the websites are hosted, "who" owns them, "how" they are managed, and "when" they were created. The given aspects work as the useful information about the hosts (there could be multiple) that can be identified by the hostname as part of the URL. The primary objective of this technique is to generate a lexical profile of the URL. This can also prevent new malicious URL from a lexical point of view to detect them unless the model of this technique is updated in order to detect new abnormal URL lexical form. It could make a profiling step for the URL in the test.

The visual similarity method is aimed at detecting targeted legitimate websites directly concerning their visual appearance. The technique works by identifying the visual similarity between different webpages. However, in cases where the phishing websites share similar visual characteristics with the authentic websites are provided, it verifies whether or not the URL is on the authentic domain. A negative response results in the identification of phishing [18, 19].

In other cases, the image is converted to a text, then it is used in some search engines and is tried to detect legitimate and phishing websites with a threshold or the PageRank [20]. Some of the features in the visual similarity process that are included in the new system are shown in Figure 2. Despite of its various advantages, one of the major drawbacks is that it may take long to load the webpage and is least effective in improving the accuracy of the existing methods [18].

Identity-based methods try to detect changes and anomalies in resource locations and their digital security, identities, and timing changes. As an example of digital security issue, consider the use of HTTPS protocol which means only that the network traffic is encrypted, while remaining uncertain regarding the authenticity of the digital certificate. Singh and Imphal [21] define identity-based antiphishing as the method which works through the validation of both online entity and user's identity during the handshake. It integrates certain credentials by client filtering and sharing technique leading towards the prevention of phishers from disguising as the legitimate entity of the website. Another example of timing issue is the age of the domain; studies integrated

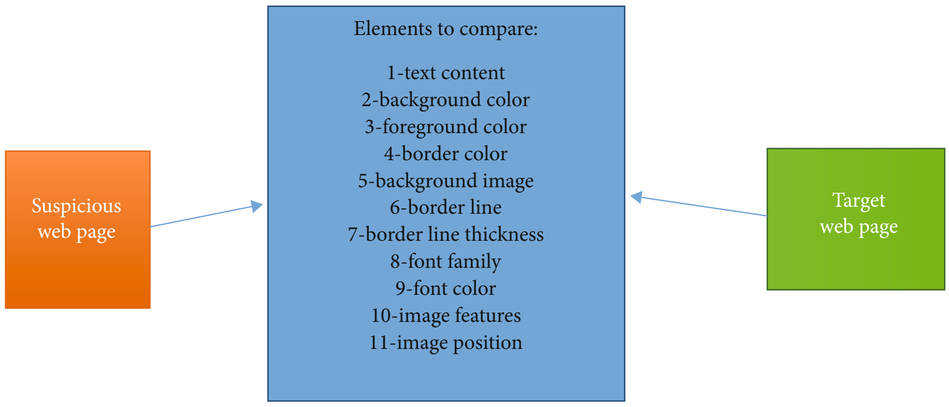

Figure 2: Visual similarity analysis process in our solution.

certain credentials by the client filtering and sharing technique leading towards the prevention and shortage of phishers' use of domain, an average of one year; generally, trusted and big companies buy domains for several years [21]. Also, some features can be gathered from the content, for example, to compare if the server where the information was filled in forms will be sent to the same server location disguising as the legitimate entity of the web page or other location or does it have the same identity or at least the same domain name or domain's name servers. URLs typically contain DNS hierarchical information, i.e., the real IP address of the domain name and hostname of the server, the path of directories and the files indicating the location of the consulted resource on the server. Generally, regular users read URLs but are not able to know that there is a DNS hierarchy that can be forged, except using "'https://." Another feature can be collected from those URLs, the age of the domain for example. Taking for example the criminology field, the identity-based method provides a profile for phishers' identity website.

Following the mutual authentication method, users are no longer in need to reenter the credentials other than the initial setup. The profiling step is an important task in the identity-based method, specifically in cases where the suspect is unknown. However, it is an abstract method because profiling identities cannot be predictable. However, identity features can be combined with profiling user's regular behaviors while using the Internet [22].

Behavioral analysis tries to identify anomalies in a given web page after executing it. User behaviors serve as the integral part of this category. For the first case, phishing web pages try to change source code after a user reacts to it by clicking on a particular button or filling forms without paying attention to the URL [23–24]. The method is also associated to the content analysis method, like trying to identify if the web page tries to execute a drive-by-download of malicious files or changing the home page of the browser. However, user's poor knowledge of internet surfing may serve as an excellent opportunity to phishers to use malicious attacks such as creating forged web pages with just a form and the other parts of the web page may contain similar images [25–26]. The above provided discussion is helpful in supporting the development of an authentic solution for phishing related problems.

3. Model and Solution Building

The system formulated in this study utilizes 37 features collected from a related but fine-tuned work using new novel rules to minimize calculation. For example, if the system detects a visual similarity from a suspicious URL that contains a known legitimate domain in its domain name, then it is directly classified as phishing. Also, if the domain was already blacklisted, then a visual similarity and page weight should be tested at first, before executing the remaining process. The way that these rules are conducted in our system is novel, but besides presenting this new way of thinking, this paper further fulfills the purpose of comparing the efficiency of using machine and deep learning model classifiers in order to determine which of them can learn better and give good accuracy and less error rate. To make it clear, the study provides information regarding the features used and logical rules conducted to classify a web page from its URL.

In the light of this, four basic requirements are considered to define the robustness, reliability, and user-friendliness of the process (Table 1). The requirements include the following.

3.1. Speed. Since phishing attacks cause considerable monetary damage few times and especially during the first hours of a phishing attack, the identification of a phish must be fast to limit the nefarious effects. Commonly, the identification of phish generally takes place during users' web surfing or email consulting. Thus, the detection method must be powerful enough to detect the phishing without affecting the quality of user's experience in the form of long-term delay.

3.2. Coverage. Phishing defenses must be able to prevent against as many vectors as possible, and a perfect phishing protection method would be able to deal with the several techniques presented in this section to provide the best protection.

3.3. Reliability. Phishing protection methods must protect the user from the most phishing attacks, as described in

Table 1: Comparison of features extraction described based methods.

Method	Speed	Coverage	Reliability	User-friendly process
Blacklisted	–	++	–	++
Lexical and host	+	++	+	++
Content	++	++	++	–
Identity	++	-	-	++
Visual similarity	++	+	+	++
Behavioral	–	++	++	-

"–" means the requirement is not satisfied; "-," the requirement is satisfied in some points; "+," the requirement is almost satisfied; and "++," the requirement is fully satisfied.

the coverage requirement. However, these must be identified to contain as low risk as possible to make legitimate communications. Too many false alarms may badly affect the user's experience. In addition, it can cause a loss of confidence in the protection technique impacting user consideration.

3.4. User-Friendly Process. Methods must be easy to use and to understand by users. Most users and especially phishing victims have few technical pieces of knowledge and few pieces of knowledge of the way phishing attacks are carried out, explaining why they are trapped. Hence, phishing protection must consider this parameter and be tailored to be easily used.

4. Engineering Features

Generally, in the field of classification problem, researches are based on preprocessed data, which means that the model can be made for specific purposes, e.g., binary or multiclassification problem; this can lead to a reasonable accuracy. The present study develops a model where 20000 active phishing URLs were collected from the PhishTank [27] platform and safe 20000 URLs from Alexa [28]. All domain names of legitimate websites are stored in a database, to analyse if there are any attempts from phishers to combine their domain name with some stop words (like sign, recovery, api, and secure) to lure victims. The dataset of the present study further involved a model with balanced labels, e.g., the number of phishing URLs should be equal to legitimate websites.

As discussed before, a total of 37 features were extracted from a URL to build model that will be a good fit to a probabilistic classifier. 85% of the threshold confidence was determined. However, if the result of the classification is superior to the given threshold, then the URL in question will be directly classified as either phishing or benign. In an alternate case, the classification was determined as follows: web page was loaded in light honey pot, to find anomalies based on predefined monitoring rules, to detect suspicious behaviors such as clicks, changing the home page of the navigator. Client's (victim's) machine was further changed by generating malicious files. Any process which involves forms to be filled with arbitrary information was also inquired to verify if the web page checks the validity of that information filled because phishing web pages generally do not make any verifications on information itself (because phishers do not have any formal database to compare the content or the structure of information provided in the target forms). If it is the case of any of those anomalies, then the current web page was classified as phishing. Finally, the overall process of features extraction was executed, and the model was updated. The provided solution may not be used as an online solution at this moment but to the researchers' knowledge and experience, it is complete and sufficient to achieve such objective, because the researchers tried to make a stress and meticulous test for URLs.

Before explaining the nature and number chosen of extracted features, the technical format of a URL was analysed as standardized in the RFC 1738 [29], which represents a resource location available on some server on the Internet.

A URL is defined as follows:

<Scheme >:< scheme-specific-part >

The scheme specifies the resource's access mechanism or network protocol communication (example: HTTP, FTP,…); the scheme selected defines the rest of the URL. For the HTTP protocol, a URL can be

<http> :// < host >:< port > / < URL_path >

It begins with the domain name or IP address of the host. At that point, there is a port number to associate it with and the URL way that gives subtle elements on how the resource can be accessed to (e.g., http://host.com:80/page); this figure gives a comprehensive example using the "@" symbol (Figure 3).

37 trusted features were selected which were divided into 6 categories such as blacklist, lexical mixed with host information, content, identity, and visual-based methods. The given features were utilized in fuzzy logic rules to construct a binary test vector which was then compared to the prebuilt model in order to get a probabilistic classification result. The features are provided by the inclusion of any one of the provided three values where 1 represents phishing, 0 represents suspicious, and -1 represents website as safe. Figure 4 and Table 2 provide the overall development mechanism of the study model and its schema in the database.

5. Blacklist-Based Method Feature

5.1. Feature 1. Black listed web name VirusTotal API [30] was used in case where the domain name was already blacklisted. The inclusion of the given software is based on its ability from different tools such as Google Safe Browsing and Avira. In cases where at least two antiviruses that consider this domain blacklisted were identified, the following considerations were incorporated: the feature will get 1 if it is the case or otherwise -1.

6. Lexical-Based Method Features

6.1. Feature 2. Feature 2 represents the IP (internet protocol): this feature evaluates if the current URL uses an IP

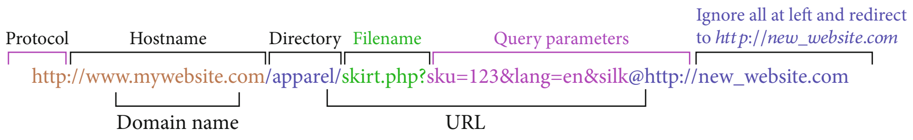
Protocol
Hostname
Directory
Filename
Query parameters
Ignore all at left and redirect to http://new_website.com
http://www.mywebsite.com/apparel/skirt.php?sku=123&lang=en&silk@http://new_website.com
Domain name
URL

Figure 3: URL standard technical structure [29].

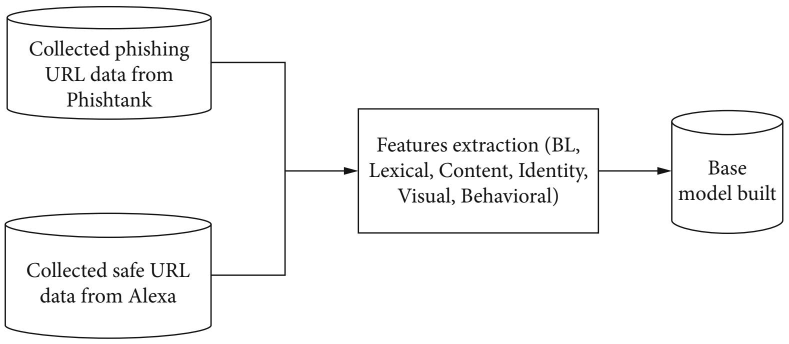
Collected phishing URL data from Phishtank
Collected safe URL data from Alexa
Features extraction (BL, Lexical, Content, Identity, Visual, Behavioral)
Base model built

Figure 4: Data collection, feature extraction, and model building.

Table 2: The schema of our model in the database.

URL	Feature1 ($f1$)	Feature2 ($f2$)	...	Feature37 ($f37$)	Result
http://www.example1.com	1	0	...	-1	1 (means it is a phishing URL)
http://www.example2.com	-1	1	...	-1	-1 (means it is safe URL)

address instead of a standard domain name. The use of IP (internet protocol) address restricts hackers to rely on DNS (domain name system) which is responsible for translating a domain to its IP address. In addition, hackers can change their IP permanently without registering it formally to avoid traceability and identity check. In our system, if the URL uses an IP address, the feature "is_ip" will have the value 1or it will get -1.

6.2. Feature 3: Lexical Length of URL. Researches showed that malicious files tend to use large URL, so that ordinary people do not pay attention to read it carefully; this can be combined with constructing bad URL format, like the use "@," "–," or "//" symbols, or making bad URL requests to servers. The given rule is defined as follows: if the length of the URL is larger than 70 characters "lexical_length_url," it will get the value 1; otherwise, if the length is between 50 and 70 characters, it will have the value 0 (0 means suspicious) or -1.

6.3. Feature 4: Use of Shortening Services. The use of shortening services is a popular approach to avoid the number of characters limited by some websites. It can be used to redirect a user to another website; it is also useful for ordinary Internet users, since this provides them the satisfaction when encountering a short URL. Canfield et al. [25] wrote a paper to show that the use of shortening may lead to a bad reputation of URL; also, it is no longer considered a good habit on cloud services. Internet users feel betrayed, as it is unavoidable that many shortening URL services do not analyse the URL in question. In the dataset of the present study, value 1 is assigned for a URL that comes from a shortening service, or else -1.

6.4. Feature 5: Inclusion of at (@). The "@" in the path of a URL redirects to what is on its right position, as shown in the URL technical specification (see Figure 3). Therefore, the use of @ symbol is a considerable direct danger, and hence, it should be regarded as disrespectful to normal internet users. In the dataset of the present study, if the URL in question uses this symbol, so "has at" will have the 1, or else -1.

6.5. Feature 6. "Double Slash" (//) Redirect. This works the same as "@," but it redirects the user into another web page in the same domain. Therefore, the feature 'double slash redirect' will get 1 in case of use, or else -1.

6.6. Feature 7: Nonstandard Port (Standard Ports: 21, 22, 23, 80, 443, 445, 1433, 1521, 3306, and 3389). This feature represents ports used for the most common standard communication to servers. Its position is just after the domain name (http://www.example.com.port). Using another port may lead to unusual communication and interaction. If that is the case, "non_standard_port" will have the value 1 or else -1.

6.7. Feature 8: Inclusion of "Minus" (-). The dash/hyphen symbol is rarely used in legitimate URLs. Phishers tend to add prefixes or suffixes separated by (-) to the domain name so that users feel that they are dealing with a legitimate webpage—for example, http://www.login-facebook.com/. The same as above, 1 will be assigned in case if it is used or else -1.

6.8. Feature 9: Number of "Dots" Present Between the Domain Name. This feature represents the number of dots present in the domain name of the URL. For example, in URL http://www.example.hud.en.ac.uk/students/, "ac.uk" is called the country-code top-level domains (ccTLD) and "example.hud.en" is the actual domain name. As the part of rule, "www." has been split to count the number of dots in the domain name (i.e., 2 dots in "example.hud.en"). Phishers tend to put many subdomains. So, if the number of dots is greater than 2, "dots_number_in_domain_name" will get the value 1, and if it contains 2 dots, it will have the value 0 or else -1.

6.9. Feature 10: Inclusion of "https" in Domain Name. Generally, browsers show URLs like "http://www.example.com/" as "http://example.com/." So, to lure normal internet users, hackers use the https in the domain name, because https represents a type of secured protocol communication with the server. Therefore, if it is the case, "https_in_domain_name" will have the value 1, or else -1.

6.10. Feature 11: Inclusion of Malicious Form. If a form in the content of the web page and the server where the information will be sent is different from the domain name or at least the name_servers of the domain name, then "is_malicious_form" will get the value of 1 or else -1.

7. Lexical and Host-Based Method (Abnormal-Based Feature)

Researches in this study extracted the source code of the web page without loading the web page itself. Python was used as a programming language, and Beautiful Soup framework to extract this source code from URL. This leads to the observation that the percentage described in some features was initiated from experience observed from phishing and safe URLs and not from a specific mathematical calculation.

7.1. Feature 12: Inclusion of Bad Request. The number of loaded objects on the web page such as images, videos, and streams was examined. Legitimate web pages generally load those objects from the same server or at least their name servers, but phishers tend to put objects from foreign servers because their techniques aim to use those objects in different web pages. A fuzzy logic was also conducted where if the external objects represent less than 25% of the total objects found, then "bad_request" will get the value -1, else if it is between 25% and 65%, it will be marked as suspicious, i.e., value 0 or else it will get a value of 1.

7.2. Feature 13: Inclusion of Anchor (#). In the source code of the webpage, redirection to another webpage is represented by the <a> tag. The number of <a> tags found in the source code was examined. It was further analysed that if the hyperlinks that were found in this tag are different from the domain name, then the percentage in the rule "bad request" will be automatically applied.

Researches indicated that phishers tend to create many <a> which points to the same web page, because their aims are not to create a functional website but to get any information filled in this web page, and this is done using techniques like <a href="#">, <a href="#content">, <a href="#skip">, and <a href="JavaScript ::void(0)">. However, the present study followed the rule, where if the percentage of such tags is less than 30% of the total of <a> found, then "Anchor" will get the value -1, or else if it is between 30% and 60%, it will be considered suspicious, i.e., 0, else the value 1 will be assigned to it.

7.3. Feature 14: Links of Tag. In HTML (HyperText Markup Language), it is probable to add some extra information to the web page by <meta> tags, or some client codes (like JavaScript) with <script> tags, or <link> to add some extra resources to the target web pages. So, like Anchor and bad request features, the study followed a rule where if the percentage of those tags that come from a domain name or name servers other than the domain name in the URL is less than 18%, "links_of_tag" will get the value -1, or else if it is between 18% and 70%, then it will be considered suspicious, or else 1.

7.4. Feature 15: Server Handle Form. This feature represents the server where information filled in forms will be sent. Generally, legitimate websites use the same domain or the same name servers, whereas phishers tend to use other malicious technics such as "about: blank" or "empty" option in <form> tags used in HTML. So, if this is the case, "server_handle_form" will be directly marked as phishing, or else if the server is different from the domain name, then it will be considered suspicious, or else it will be safe.

7.5. Feature 16: To Email. This feature describes the fact that phishers send information filled in forms by emails using "email:to" option in <form> tag, which is very suspicious. If this is the case, the feature will be marked as phishing or else safe.

7.6. Feature 17: Abnormal Structure. Legitimate websites generally use the same identity in the domain name and URL in the request; the identity can be gathered using WHOIS information. This is not the same for phishers who try to hide their identity. So, if that is the case, "abnormal structure" will be considered phishing or else safe.

7.7. Feature 18: Punycode. The use of this technique is new, and it is a system for converting words that cannot be written in ASCII (American Standard Code for Information Interchange), such as Ancient Greek. The phrase ΓΝΩΘΙΣΕΑΥΤΟΝ ("know yourself"), once converted into ASCII characters, looks like this: xn–mxadglfwep7amk6b. Another example is this URL "https://xn–80ak6aa92e.com/"; it will be translated as such "https://www.apple.com/." If the browser used is not protected against this attack, it will be almost impossible to detect. This is why it is included in the system of features within the proposed solution. So, if that is the case, the feature will get the value 1 or else -1.

8. Content-Based Method Features (HTML and JavaScript-Based Features)

8.1. Feature 19: "Redirect." Researches showed that legitimate websites make redirections before arriving at the URL in question, but phishing can use the redirection for more than three times. So, it was deduced that if the redirection was made one time, the feature "is_redirect" would be marked as safe; or else if it was made two or three times, it would be marked as suspicious, else it will be considered a phishing threat.

8.2. Feature 20: Mouse on over. JavaScript permits handling events such as the mouse and keyboard. The option of "onMouseOver" can change the status bar of the current web page; this can help phishers to show a fake URL on the status bar. So, if this is the case, this feature gets the value 1 or else -1.

8.3. Feature 21: Favico. "Favico" represents the favicon of a website; if this favicon is loaded from another server other than where the web site is stored, then the feature will attain the value 1, or else -1.

8.4. Feature 22: Right Click Disabling. If the website does not permit the right click option, then this feature will be provided with value 1 or else -1. This option for example can help to prevent the user from checking the source code when it is obfuscated.

8.5. Feature 23: Pop-Up. It is probable to check if there is a pop-up window that asks for personal information without any suspicious reasons. Generally, legitimate websites use a pop-up to ask for emails to warn about something or announcement. This feature will get the value 1 if a pop-up is used, or else -1.

8.6. Feature 24: iframe Use. An iframe can be represented as a website inside a website. iframes can have the possibility to be hidden. This gives rise to the possibility of threat of using hidden iframes, as gathering victim's machine information or using them as keyboard key logger.

An example of HTML code to create a none visible iframe is as follows:

```
<iframe src="http://malicious.com/index.php" width= "1j" height= "1j" style= "visibility: hidden;">
</iframe>
```

9. Security and Identity Method Features (Domain Based Feature)

9.1. Feature 25: Age of Domain. Legitimate websites purchase domain names for at least one year or sometimes more, but for phishers who aim to execute their attacks for a short period, their domain age is approximately six months. This information can be collected from the WHOIS database. So, if the domain name of the URL in the test is less than six months, then "age of domain" will get the value 1 or else -1.

9.2. Feature 26: ssl (Is the Digital Certificate Trusted?). Using HTTP as a protocol of communication between a user and a server only means that the network is encrypted, but that does not mean that the used digital certificate can be trusted. Even if phishers have a trusted digital certificate, its age is almost in the average of a one-year lifetime. Legitimate websites often buy trusted digital certificates for many years. Considering this, the following rules have been extracted: if the web page in question uses a trusted digital certificate and has more than one year as lifetime, then it will be regarded as legitimate. Otherwise, if it is not trusted and has more than one-year lifetime, then it will be seen as suspicious, or it will be considered phishing.

9.3. Feature 27: dns. In the WHOIS database, it was checked if there are any DNS records for the URL in question, if that is the case, this feature will be represented as -1 or else 1. However, in usual conditions, phishers generally tend to hide their identity.

9.4. Feature 28: Popularity. This feature measures the popularity of a website; it is calculated by the number of visitors. Generally, phishing websites are not heavily visited. WHOAPI is a web service that gives website popularity which helps us to get this information. Researchers observed that legitimate websites have a popularity of more than 100000 (lesser is better; number 1 is Google) which will be our threshold. The rule will be based on this number.

9.5. Feature 29: Rank. Website rank is their importance calculated by the PageRank algorithm. If the website in question has a rank of more than 25%, it will subsequently be considered legitimate and will be represented as -1; otherwise, the website is termed as phishing and is ranked 1.

9.6. Feature 30: Index (Is It Indexed in Search Engines). If the current URL appears in a search engine like Google Index, it will be legitimate or else phishing.

9.7. Feature 31: Page Links (Number of Pages Pointing to This Page). Generally, legitimate websites have other websites that point to them; this gives an impression about their legacy. Phishing websites do not have this option due to their short lifetime. So, if there are no links that point to the URL in question, it will be marked as phishing, if the number is between 0 and 2, it will be seen as suspicious or else as legitimate.

9.8. Feature 32: Statistics Report. PhishTank is an organization that specializes in detecting phishing websites that appear daily, so if the domain appears into the PhishTank database, it will be determined whether it is phishing or legitimate.

10. Visual Features of Webpage

This part consists of comparing two web pages from a visual appearance point of view. All the domain names of legitimate websites were collected and stored by the researchers. If the URL uses a domain name like http://login-facebook.com for example, then it is logical to start by a visual comparison between the screenshot of this URL and login page of http://facebook.com. Generally, phishers target the login page of legitimate websites. To do so, some defined useful features have been defined to be used such as {text_content, background_colour, foreground_colour, border_colour, background_image, border_line, border_line_thickness, font_family, font_colour, image_features, image_position} (see Figure 2).

There are 11 features in total. In cases where change in any feature were detected (for example, the text content in the URL in the test is the same as the target, so "text_content" will have the value 1), it will either have the 1 or -1. Then, the number of ones were calculated and were divided by 11; if the result is more than 60% of resemblance, then the URL will be directly marked as phishing URL. This led towards the process of feature extraction to update the model of this study.

11. Behavioral Features of Webpage

As mentioned before, the process of feature extraction is performed in static analysis which means that the webpage was not loaded. The completion of the process resulted in the provision of the binary test vector, i.e., [-1, 1, 0, -1, 0,,1], which was further injected into a classifier to provide a probabilistic classification result where for instance a URL with a probability of 95% is referred to as phishing. However, in some cases, the system may give us a probability of under 85% (the threshold of confidence), so a human verification to the URL was undertaken by analysing its behavior as follows:

A Linux distribution with full tools was configured to analyse the network traffic, with the aim at letting the website run freely. The web page of the URL was loaded in order to answer the following questions: (i) does it change the home page of the browser? (ii) Does it propose any notifications? (iii) Does it download some files directly after loading the page and without the users' interaction? (iv) Does it try to change the OS registry? Etc. In cases where any of the malicious behaviors was found, the targeted legitimate websites were then considered while reproducing and updating the overall system and model.

12. Static, Dynamic, and Hybrid Analysis

Static analysis refers to detecting an error without executing the main target. It helps in general because the static analysis does not depend on the software or hardware to execute this task. In the present study, static analysis refers to extract the source code from a URL without executing it to be fast in procedure treatment. However, a problem can sometimes be encountered, such as some web pages generate encrypted source codes from both legitimate or phishing web pages that can be solved because there are some tools to restore this source code of the web page. The purpose of the static analysis is to find specified keywords or fragments directly.

Likarish et al. [31] extracted 65 features of JavaScript code and established several classifiers to detect malicious JavaScript code of web pages. Besides feature selection, some irrelevant features can be found, like reserved words in JavaScript. Also, the features of the HTML source code content of web pages were also extracted for machine learning classifiers in [32, 33]. The system proposed in the present study used different types of features discussed above that can be easily extracted from the source code; this helps us be more effective and faster.

Dynamic analysis needs specific resources (software or hardware) to work with because executing targeted tests needs an appropriate environment. Dynamic analysis is more straightforward than static ones since it executes the source code of the webpage, and intriguing occasions are observed and logged after execution [26, 34–36]. According to its different implementation platforms, dynamic analysis can be classified into three main groups, i.e., low, medium, and high interaction honeypots. Features extracted from this technique help improve behavioral and semantic features of suspected webpages. In the present study, a Linux distribution is configured with preconfigured specific tools for network monitoring and logging to deal with behavioral features described in part 8 from feature engineering. Hybrid analysis is the fact that a system uses both static and dynamic analyses. Similarly, the solution proposed in the present study was undertaken by the process of static analysis; in cases where the results were not convincing, dynamic analysis was undertaken. By analysing a total of 37 URLs, comprising of 34 phished URLs obtained from the PhishTank and 20 legitimate URLs, 14 features of an URL were defined to distinguish phished one and legitimate one.

Heuristic 1 is the length of the host URL. A URL is a string that contains information used to recognize a specific website that that URL is assigned to. Important parts of a URL are network protocol, host name, and path. The domain name length is examined in this heuristic. The average length of phished URL is more than 25 characters, while the average length for legitimate URL is 20 characters. Heuristic 2 is the number of slashes in URL. To trick people to believe in a phished URL, slashes are usually added to the URL to subtly distinguish an imitation from the imitated. The study found that if number of slashes in URL is greater or equal to five then it is phished URL. The number of slashes in legitimate URL is usually three. Heuristic 3 is the number of dots in the host name of the URL. This heuristic found that more dots are used in phished URL to make people think it is a legitimate URL. So, if the number of dots in host name is more than four, then it is phished URL. Only if the number of dots is less than or equal to three, then it is legit. Heuristic 4 is the number of terms in the host name. The terms in host name in a URL are tokenized to make the URL not containing any sensitive information. Here, the number of terms in host name is examined. If it is greater than four, then it is phished URL; the average number of terms is four in the legitimate URL.

In the heuristic number 5, special characters are considered. If the appearance of special character is found in a URL, then it is phished URL. In the process of extracting features, the paper found that 77.75% of phished URL contain special characters. Heuristic 6 is the IP address. The domain name in a URL is mostly used to address a legitimate website, and if a URL contains an IP address, then it certainly is a phished URL. In the dataset being examined, 9.4% of phished URL have an IP address in their URL. For heuristic 7, we consider Unicode in URL. The paper found that if there is Unicode in the host name of a URL, then it mostly is a phishing URL. 65.16% of phished URL are found with Unicode. Heuristic 8 is the transport layer security. The transport layer security is used to protect sensitive information in a website, and it also is used as an indicator for legitimate URL. In the process of extracting, it is found that 99.16% of phished URL do not have transport security layer.

The ninth heuristic examines the subdomain of a URL. Internet users usually mistake that more subdomains in a URL means that that it is legitimate. However, the paper found that 64% of phished URLs are with subdomain. Heuristic 10 is some certain keyword in the URL. If some common keywords appear in the path portion of a URL, it is a phished URL. The paper found that 91.08% of URLs have certain keywords in the URL. Heuristic number 11 is the top-level domain. In the host name of a URL, there are top-level domain, secondary-level domain, and the domain. If a URL does not have the top-level domain, then it is a phished URL, and it is discovered that 66.5% of phished URLs in the database do not have top-level domain. In the heuristics number 12, the number of dots in the path of the URL is considered. If the number of dots in the path of the URL is found to be greater than two, then it is a phishing URL. On the other hand, the number of dots in the path of legitimate URL is less than two. Heuristic 13 indicates that if the appearance of hyphen in the host name of the URL is more than one, then it certainly is a phished URL. In the legitimate URL, a hyphen only appears once or less. The last heuristic 14 examines the URL length. It is found that a phishing URL has a length greater than 75, while the length of legitimate URL is about 40.

13. Dimensionality Reducing Features

The process of features extraction led to a binary vector of 37 features, where each element can have only three possible values {-1, 0, 1}. This vector might be compared with a prebuilt mathematical model. This model contains an additional feature called "Result" which determines if a current vector represents a phishing one with the value 1 or benign with the value -1. In the worst case scenario, all the 37 features were extracted to be trained in the model, but mathematical researches showed that it is possible to get almost the same accuracy value using a limited number of features, like using only 25 features out of 37, using algorithms like PCA (principal component analysis), RFE (recursive feature elimination), and UFS (univariate feature selection). Feature selection is divided into three main categories: filter, wrapper, and embedded methods [37].

PCA is useful in finding similarities and differences between features. Since patterns can be hard to find in data of high dimension, it is a system used to underscore variety and bring out solid examples in a dataset. It makes data easy to understand, explore, and visualize. The point of principal components analysis (PCA) is to lessen the dimensionality of an arrangement of factors while holding the greatest changeability as far as the fluctuation covariance structure. In other words, PCA endeavours to clarify the change covariance structure of an informational feature utilizing another arrangement of facilitating a system that is lesser in measurement than the number of unique features. Given an arrangement of M features, say X, a principal component (PC) show changes these features into another set lesser in measurement, i.e., $C < M$, but can catch the more significant part of the fluctuation in the first informational collection. Each facility in the new changed system is known as an essential part and hence the name principal component analysis [38].

RFE (recursive feature elimination) points are used additionally to lessen the number of features used to build models before fitting them to the classifier. RFE, as its title recommends, recursively expels highlights, fabricates a model utilizing the rest of the qualities, and calculates model accuracy. RFE can work out the mix of attributes that add to the forecast on the objective variable (or class). It has a category with wrapper strategy methods. At first, the calculation fits the model with regards to all the given indicators. Every indicator is positioned utilizing its significance to the model, given that "S" is a chance to be an arrangement of requested numbers which are competitor esteems for the number of indicators to hold ($S1 > S2,\ldots$). At every cycle of highlight determination, the Si top-positioned indicators are held, the model is refit, and execution is evaluated. The estimation of Si with the best execution is resolved, and the best Si indicators are used to fit the last model [39].

UFS (univariate feature selection) measures the significance of each element which is independent from anyone else. For instance, it includes the best segregation between the states of intrigue exclusively (i.e., univariate wrapper technique) that can be chosen. It unavoidably disposes of highlights that, when taken in total, would have given helpful data about the test conditions [40].

In this paper, two common association rule mining techniques are used: apriori and predictive apriori. The purpose of association rule mining is to find the association between sets of items in the database. The association rule mining uses the support and confidence calculated from the number of item's appearance to generate the rules. Support is the number that an item appears in the database. Confidence is the conditional probability of an item A with reference to item B, calculated by the number of times that A and B appear together, divided by the number of appearances of A. Though apriori and predictive apriori algorithms have some similarities, the major difference is that the apriori algorithm uses confidence to generate rules, and the predictive apriori algorithm uses confidence and support together. They are combined in a measure called accuracy, and predictive apriori algorithm uses accuracy to generate rules.

Figures 5 and 6 present a sample of the developed model after feature extraction, and Figure 7 provides a comparison between PCA, RFE, and UFS which is useful in reducing the number of features, trained with SVM (support vector machine) classifier and test the accuracy (accuracy means how many correct predictions were made from all the

is_ip	lexical_length_url	using_shortening_service	has_at	double_slash_redirect	has_minus	dots_number_in_domain_name	ssl	Domain_registration_length
−1	1	1	1	−1	−1	−1	−1	−1
1	1	1	1	1	−1	0	1	−1
1	0	1	1	1	−1	−1	−1	−1
1	0	1	1	1	−1	−1	−1	1
1	0	−1	1	1	−1	1	1	−1

main_name	ssl	Domain_registration_length	favico	...	pop_up	iframe_use	age_of_domain	dns	popularity	rank	index	page_links	statistics_report	result
−1	−1	−1	1		1	1	−1	−1	−1	−1	1	1	−1	−1
0	1	−1	1		1	1	−1	−1	0	−1	1	1	1	−1
−1	−1	−1	1		1	1	1	−1	1	−1	1	0	−1	−1
−1	−1	1	1		1	1	−1	−1	1	−1	1	−1	1	−1
1	1	−1	1		−1	1	−1	−1	0	−1	1	1	1	1

FIGURE 5: Description of the built model (sample of the dataset).

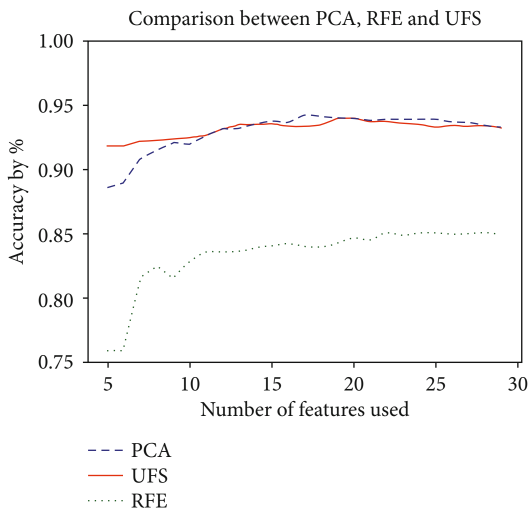

FIGURE 6: Comparative analysis of PCA, RFE, and UFS.

predictions made). Therefore, numbers from 5 to 30 features are used.

It helps to reduce the number of features to be fitted into the model. The number of features is chosen between 5 and 30 from our model. This figure shows clearly that even a small number of features can give excellent accuracy in comparison to using all features, PCA at 17 features and UFS at 20 features.

Because two different algorithms are used, the rules generated from them are different. While apriori algorithm extracts rules that contain most common features like subdomain, URL without transport layer security, and keyword in the path portion of the URL, predictive apriori generates other rules that are widespread and contains more features, so it is easier to indicate phished URL. Another thing to compare is time used to generate rules. Apriori is much faster than predictive apriori especially when more instances are considered. However, predictive apriori rules are considered for further process while apriori rules are not. An interesting fact is that when 100% of the input dataset is used, 31 unique rules are mined by predictive apriori. Meanwhile, when 10% of the input data set is mined using apriori algorithm, 27 best unique rules are generated. That means predictive apriori algorithm is much more effective when we use the whole database, while apriori algorithm is only effective in generating rules when a small amount of database is used.

14. Machine and Deep Learning for Building Classification Models

14.1. Machine Learning. Artificial intelligence (AI) is a territory of software engineering that underscores the making of insightful machines that work and respond like people. A portion of the exercises PCs with AI reasoning is intended to include speech recognition, learning, planning, and problem explaining. Machine learning is an aspect of AI, where PCs figure out how to determine an issue without human intervention, and this is through gaining information. Therefore, high-level information would result in the development of quality machines. Machine learning centres on the advancement of PC programs that can get to information and use it to learn for themselves.

Studies related to the detection of phishing use machine learning as a way to learn from collected data. Specific focus is granted towards the approaches that are useful in analysing information coming from a URL and its corresponding webpages, by extracting good and reliable features, then submitting them as a classifier like CART (decision trees), KNN (K-nearest neighbors), or SVM (support vector machine).

14.1.1. CART (Decision Trees). A decision tree is a predictor, $h : X \longrightarrow Y$, that predicts the name or label related to an instance or a case X by going from a root node of a tree to a leaf. It can be used for binary and multiclassification problems. At each node on the root-to-leaf path, the successor child is chosen based on a splitting of the input space. Usually, the splitting is based on one of the features of X or a predefined set of splitting rules [41].

14.1.2. KNN (K-Nearest Neighbors Classification). KNN classification is an instance-based supervised learning method that works well with distance-sensitive data. It suffers from the curse of dimensionality and other problems with distance-based algorithms. The idea is to remember the training set and afterward foresee the mark of any new

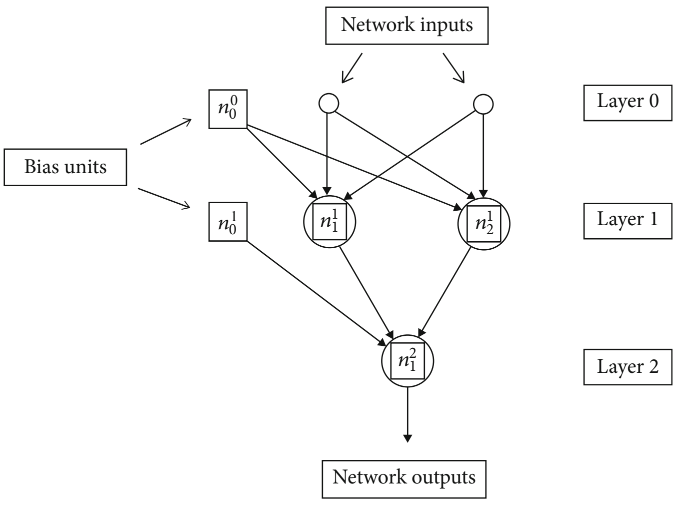

Figure 7: Example of MLP architecture with one hidden layer.

example based on the names of its nearest neighbors in this set. The rationale behind such a method assumes that the features that are used to describe the domain points are relevant to their labeling in a way that makes close-by points likely to have the same label [42].

14.1.3. SVM (Support Vector Machine). SVM is used for learning linear predictors in high-dimensional feature spaces. The high dimensionality of the features space raises both sample complexity and computational complexity challenges. The SVM algorithmic paradigm tackles the sample complexity challenge by searching for "large margin" separators. Roughly speaking, a half space separates a training set with a large margin if all the examples are not only on the correct side of the separating hyperplane but also far away from it. Restricting the algorithm to output a large margin separator can yield a small sample complexity even if the dimensionality of the feature space is high (and even infinite) [43]. Depending upon the above information, the current study utilizes the above three classifiers, since they are heavily used due to their quality performances [44–47].

14.2. Deep Learning. A neural system is a model of calculation roused by the structure of neural systems in the cerebrum. The disentangled models of the cerebrum are comprised of countless registering gadgets (neurons) that are associated with one another in an unpredictable correspondence organization, through which the mind can complete profoundly complex calculations. Counterfeit neural systems are formal developed calculations that are designed according to these calculations worldview. Learning with neural systems was proposed in the Mid-Twentieth Century. It has been shown to achieve very impressive performances. A neural system can be portrayed as a coordinated chart whose nodes relate to neurons and edges to compare between them. Every neuron gets a weighted sum of the neurons' yields associated with its rising edges, in the form of information. This paper utilizes two models of neural networks such as MLP (multilayer perceptron) and CNN (convolutional neural network). Deep learning for phishing detection is a growing domain which has also attracted the interest of different researchers [48, 49].

14.2.1. MLP (Multilayer Perceptron). The multilayer perceptron is the most widely used model of neural network. A significant part of the notoriety of MLPs can be owed to the way that they have been connected effectively to an extensive variety of data undertakings, including design grouping, workplace learning, and time arrangement forecast. Practical applications for MLPs have been found in such diverse fields as speech recognition, image compression, medical diagnosis, autonomous vehicle control, and financial prediction, and new applications are being discovered all the time [50]. MLPs are trained, rather than programmed, to carry out the chosen information processing task. MLP training involves the adjustment of the network so that it can produce a specified output for each of a given set of input patterns. Since the desired outputs are known in advance, MLP training is an example of supervised learning. The MLP architecture consists units or nodes arranged in two or more layers (the input layer, which serves only to distribute the input from each pattern, is not counted). Real-valued weights connect some of the nodes, with no connections between nodes in the same layer.

14.2.2. CNN (Convolutional Neural Network). CNN, or ConvNets, is quite similar to regular neural networks discussed above. They are still made up of neurons with weights that can be learned from data. Each neuron receives some inputs and performs a dot product. Despite everything, they have a misfortunate work on the latter that completely associates with the layer which means that they use a loss function too. They can, in some cases, utilize a nonlinearity function.

As mentioned before, a consistent neural network gets input information as a solitary vector and goes through a progression of hidden layers. Each hidden layer comprises an arrangement of neurons, wherein each neuron is completely associated with the various neurons in the

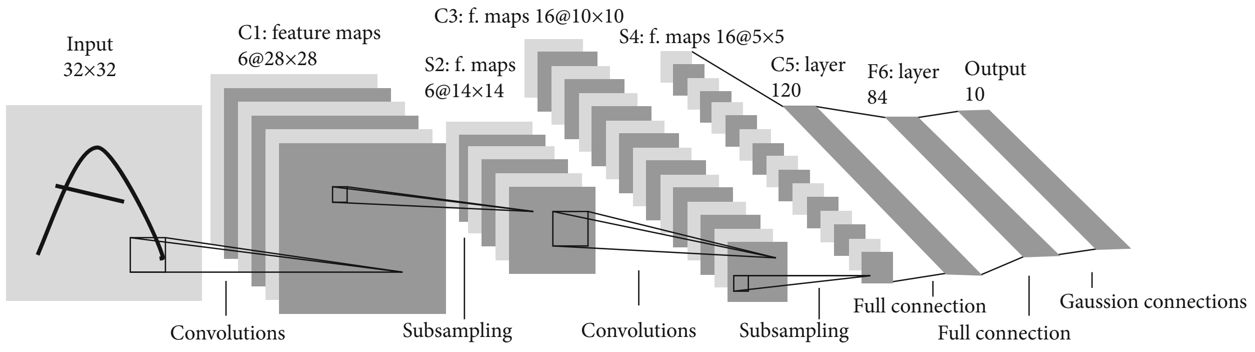

Figure 8: CNN architecture example for character recognition [52].

Table 3: Comparison of advantages and disadvantages of different phishing detection techniques.

Methods	Advantages	Disadvantages
Blacklist	Does not require high resources to use it Fast Excellent when minimal FP (false positive) required	Not trusted Not always updated
Lexical and host	Provides a sort of lexical profile for such URL Can prevent and not only detect	Should always be updated Needs human intervention sometimes
Content	Trusted Detect hidden content like iframes	Cannot detect obfuscation Should be maintained
Identity	Offers an owner profile	Not trusted
Visual similarity	Effective Mitigate zero-hour attack	Higher FP rate High computational cost
Behavioral	Detects hidden anomalies Reveals novel abnormal behaviors	Very high computational cost Needs human intervention Should be maintained
Machine learning	Can prevent and not only detect Mitigate zero-day attack Construct its own models	Time-consuming Needs maintenance Works through several rules
Deep learning	Can prevent and not only detect Mitigate zero-day attack Constructs its own models Becomes more effective with data increase	Time-consuming Should be maintained A vast number of rules Massive number of parameters to handle

previous layer. Within a single layer, each neuron is completely independent, and they do not share any connections. The last fully connected layer, also called the output layer, contains final class scores. Generally, there are three main layers in a simple CNN model: the convolution layer, the pooling layer, and the fully connected layer (Figure 8). CNN differs from MLPs in the types of hidden layers that can be included in the model. A CNN model arranges its neurons in three dimensions: width, height, and depth. Each layer transforms its 3D input volume into a 3D output volume of neurons using activation functions [51].

Table 3 represents the advantages and disadvantages of different phishing detection techniques.

In the next section of experimentations and results, other concepts from deep learning as training or loss functions, "Regularizes," and dropout will be explained to understand the goal of fine-tuning and optimizations of the model developed in this study.

15. Model Development and Conducting Logical Rules

After the data collection process, this study extracted reliable features using six based methods using static analysis to be possibly completed in order to construct some profiles for the URL. After that, a dimensionality reduction of features was carried out to try to get the same final result of classification using fewer calculations (for example, only 30 of 37 features will be used). Then, this vector was trained with a probabilistic machine learning or deep learning classifier after fine-tuning its hyperparameters to get the best result of the classification process. When that is finished, the whole process including blacklist, visual similarity process, static feature extraction (based on methods like lexical, content, and identity), and behavioral analysis was executed again independently to update the designed model and to answer the question of how phishing phenomenon evolves and what the new most effective and used features are.

The CNN model has shown a perfect accuracy of 97.945% and a little error rate (false positive and false negative detection) of 2.1%. However, the results are not to be trusted 100%. We studied in our research what have been accomplished by recent researches and tried to get the most relevant features from different based methods used in binary classification in the field of phishing URL detection, all with the sole objective of being as complete as possible. Then we tried to build the model from a top level, in a way that we collect about 20000 active phishing URLs.

The way towards learning starts with perceptions or information, for example, precedents, coordinate involvement, or guidance, keeping in mind the end goal to search for examples in information and settle on better choices later on in view of the models. The essential point is to permit the PCs to learn consequently without human intercession or help and to alter activities as needs be. In general, there are three fields of machine learning or to put it simply, "the way machines gain from perceptions," for example, administered, unsupervised, and reinforced learning. On the other hand, we can rely on the conducted rules not to perform unnecessary treatments; however, the whole process should be executed at the end in order to update the model used with new tested URLs to get more knowledge about new technics and methods used by phishers.

The main aim of this article is to show how we can optimize a system by selecting the most trusted features by feature selection, dimensionality reduction engineering, and fine-tuning the hyperparameters of the machine and deep learning models. An overfitting case is one where execution on the training set is excellent and keeps on enhancing, while execution on the validation set enhances to a point and after that starts to debase. Secondly, those chosen features come from the fact that we want to be as complete as possible. So we believe that a stress test on URLs from a different kind of analysis is good enough rather than using a single mode of analysis such as blacklist, lexical, or another kind of analysis. Thirdly, we prefer that our system carries out a static analysis at first rather than dynamic not to be dependent on the software or hardware needed in the execution of the web page in question.

This study presented a novel hybrid solution based on fuzzy logical rules as a new technique of detecting phishing URLs, where static analysis was used as base treatment: visual comparison or features extraction depending on the URL domain name was already blacklisted for the domains which contained the legitimate domain names database. In other complicated cases, the process was undertaken through the dynamic analysis by analysing the behaviors and the content of the web page belonging to this URL. This may give rise to questions regarding the usability of solution, its functioning either on batch mode or the online mode.

To cater the above complexities, the article provided a trusted solution which is used to detect and prevent phishing web pages, especially for batch mode use. All the experimentations were made in a single machine with i7-6200 octa-processors, 16 GB of RAMs, type DDR4 and SSD hard disk drive with parallel tasks execution option to be as fast as possible.

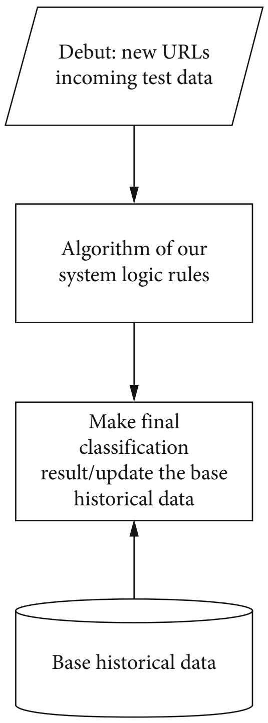
Debut: new URLs incoming test data
Algorithm of our system logic rules
Make final classification result/update the base historical data
Base historical data

Figure 9: The general process of our system for phishing detection from URL.

Since deep learning networks need a lot of resources, if the user intends to use this in an online mode, it should be scaled with huge machine resources in the cloud. The major challenge of using a probabilistic binary classification in this study was that at the end of the process, researchers intend the output value which contains a probabilistic prediction regarding URL of being phishing or benign. The detailed output of the result is described in Figure 9.

A threshold of confidence was defined as 85% for the static analysis process, which is a very high threshold. However, during the experimentation process of this study, there were times when the researcher encountered a prediction under this threshold. This is due to the development of new powerful rules to classify URLs and to reduce computation time. In addition, the visual similarity approach was efficient enough to detect targeted legitimate websites easily. Features extracted are built from literature and personal experiences. Firstly, a vast amount, i.e., around 40000 of active URLs were collected from PhishTank and Alexa, where 74% were safe URLs and the remaining 26% URLs were affected through phishing. The model utilized ought to be adjusted which implies that the quantity of phishing and safe URLs is equal to abstain from an overfitting problem.

However, the selection of the feature is based on the fact that the model must be complete and efficient with the functional point of view. Figure 10 provides the general flow chart of system analysis from their URL, as performed in this study;

Figure 11 provides the complete functionality of the newly developed system. To provide results with increased efficiency, the researchers made sure to use fresh data only.

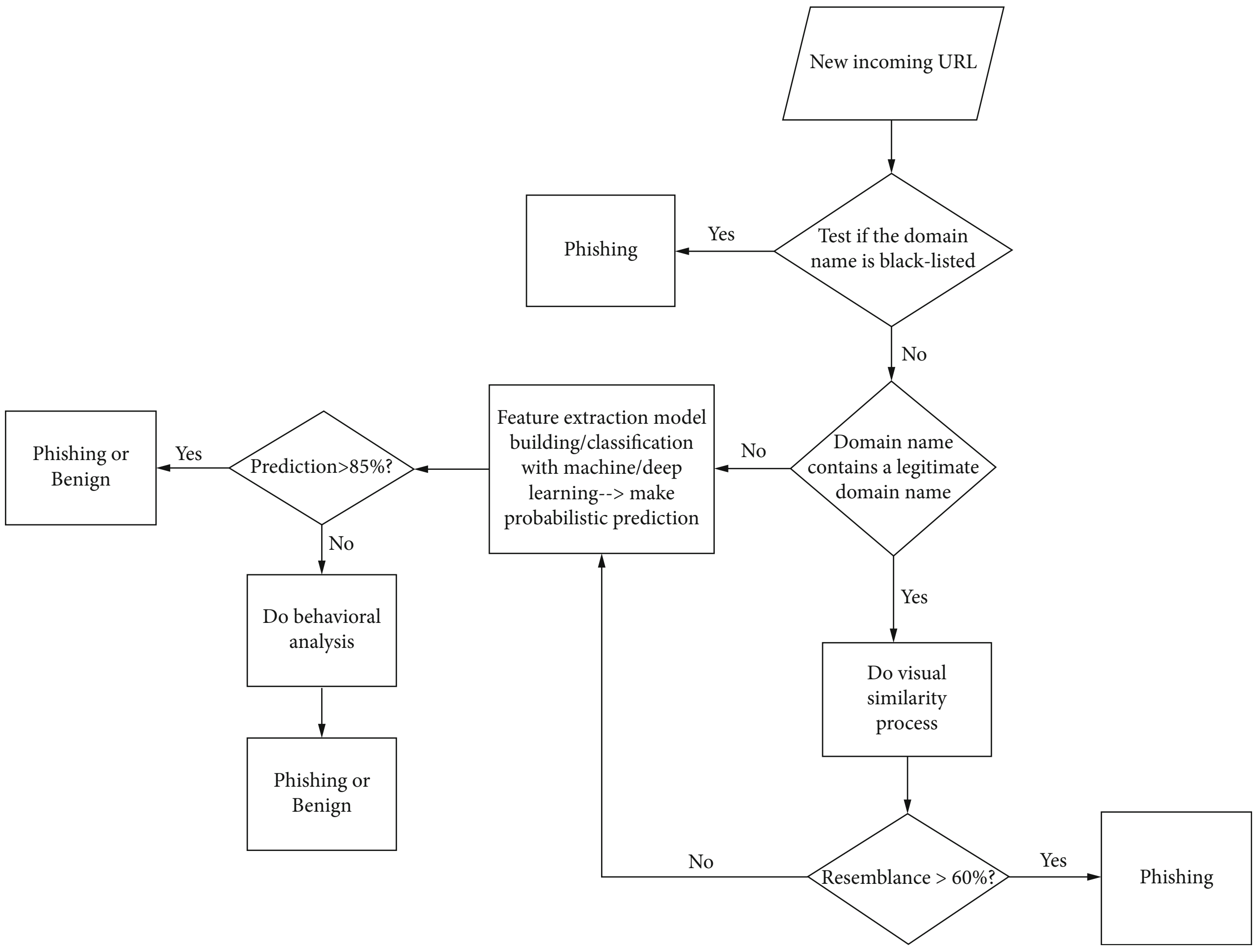
New incoming URL
Test if the domain name is black-listed
Yes
Phishing
No
Domain name contains a legitimate domain name
No
Feature extraction model building/classification with machine/deep learning--> make probabilistic prediction
Prediction>85%?
Yes
Phishing or Benign
No
Do behavioral analysis
Phishing or Benign
Yes
Do visual similarity process
Resemblance > 60%?
No
Yes
Phishing

Figure 10: Our system logic rules.

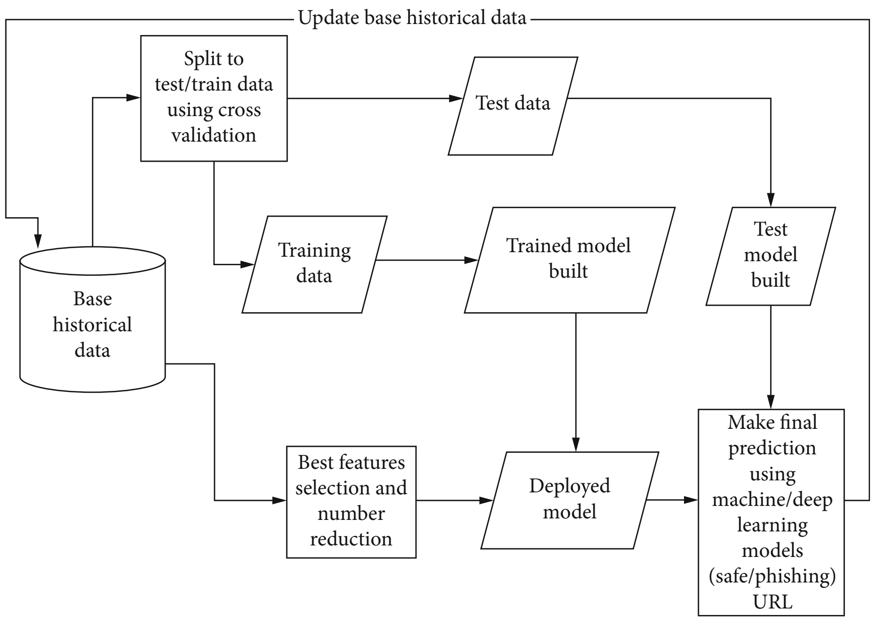
Update base historical data
Split to test/train data using cross validation
Test data
Training data
Trained model built
Test model built
Base historical data
Best features selection and number reduction
Deployed model
Make final prediction using machine/deep learning models (safe/phishing) URL

Figure 11: Machine and deep learning classification process flow chart.

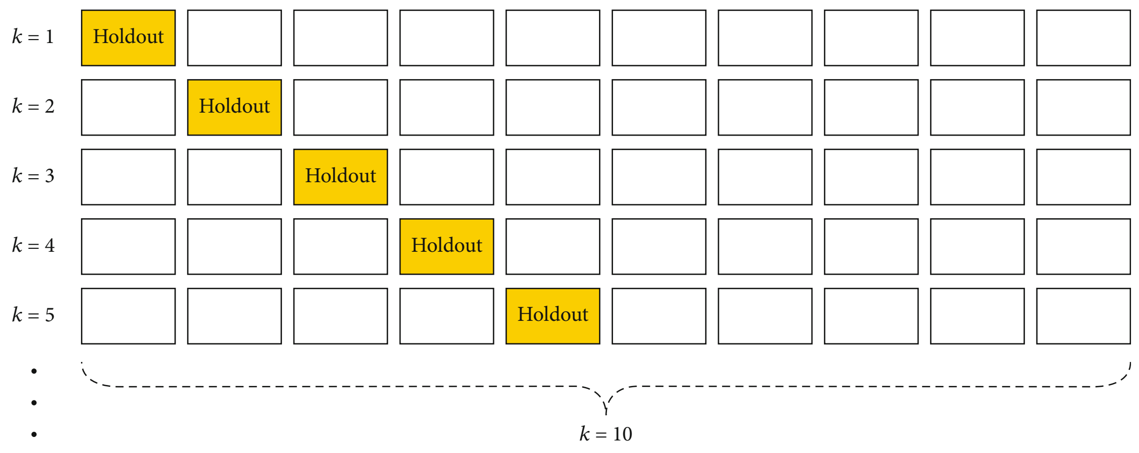

Figure 12: Cross-validation.

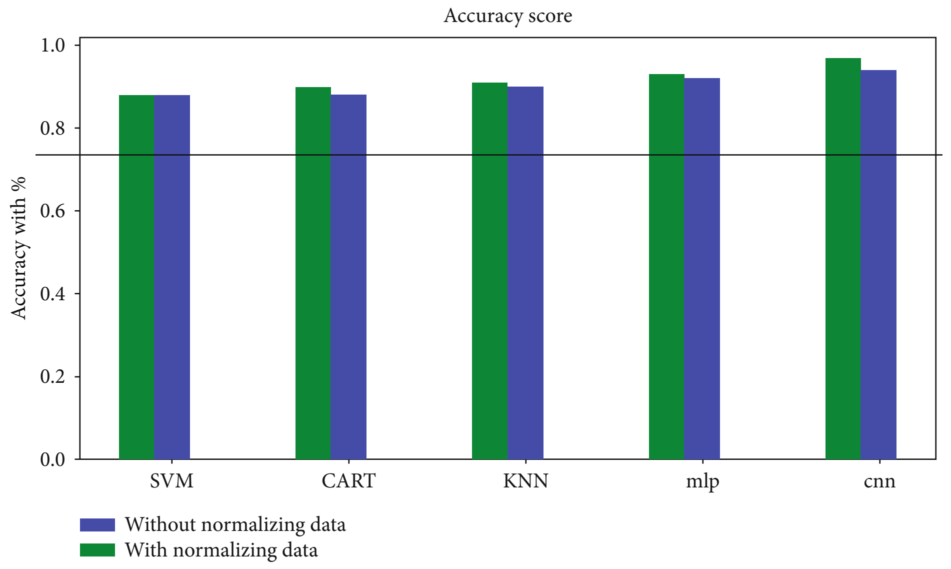

Figure 13: Comparison of accuracy performance between all algorithms tested with and without normalizing data.

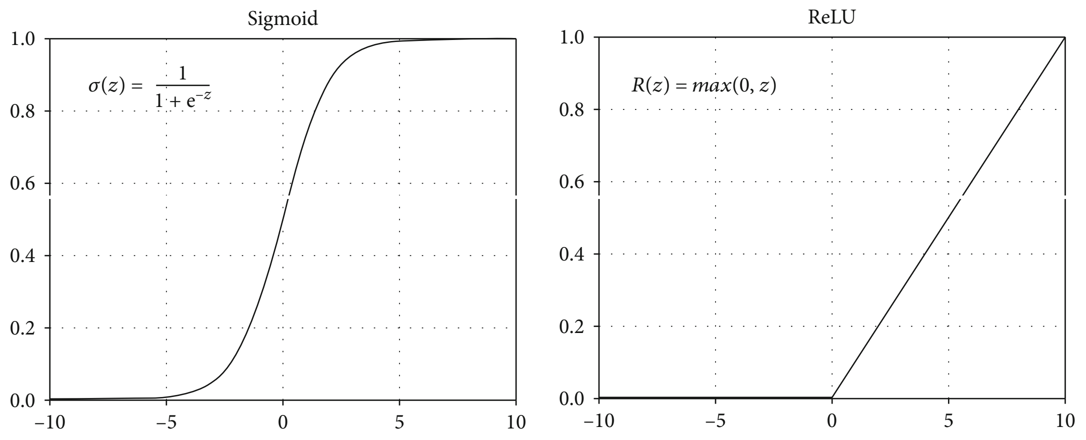

Figure 14: Sigmoid and RELU activation functions.

Certain new rules were developed to avoid some unnecessary treatments. Flow chart helped us in 72% of all the tests data. In 100 case studies, the rules were valid for 72 of the cases. The algorithm is useful as it can detect phishing URLs without the need to go through the whole process of extracting features, building and training model, and at least making predictions. Also, behavioral analysis helped researches with 15% of the case studies.

16. Model Training

There are many ways to build and train a model; however, in the present study, a cross-validation as a manner to do it. Cross-validation is a technique for getting a dependable gauge of model execution utilizing just the training data. It splits the whole data in two parts, one for training data (used to learn) and the other for test data (used to perform the learning before moving onto the classification process of the new test data). A percentage of 80% was defined to be trained and 20% to be test data.

Python was used as a programming language due to its simplicity and rapidness to write codes. In addition, Scikit-learn library [53] was also used to perform cross-validation and models building with machine learning, and the library Keras [54–55] to build models with deep learning networks. Scikit-learn and Keras are very popular libraries to perform data analysis because they are reliable and scalable.

Our data is a tabular data formatted as a CSV file. It contains 40000 of a binary vectors with 38 (37 features are extracted, and the last one is the feature "Result"; see Table 2) features representing phishing data and the same for benign data. It contains the correct answer whether such vector belongs to phishing or benign with the feature "Result."

In order to get a reliable model, cross-validation was used where data was split tenfold with randomly chosen lines, and each one will use the others as test data (Figure 12).

The data is split into ten parts so that the model building process will be executed ten times, and each time, the hold-out will be used as test data. When the data is split, it will be standardized (normalized) to have equal probabilities for the prediction process. It is requested to get better performance for such algorithms like KNN and SVM and to reduce calculation in CNN architectures. The StandardScaler package was used from Scikit-learn. As an illustration, this figure represents the difference using accuracy (number of correct predicted values divided by all instances).

It further involved the deep learning networks to test the efficiency of the provided solution. The multiple layer approach of deep learning was used. The objective behind using multiple layers is to help the network use some additional functionalities that classic machine learning algorithms do not have like activation function, which is responsible for learning from previous outputs that come from a layer to another one (Figure 13). This function helps to basically decide whether a neuron should be activated or not. Whether the information that the neuron is receiving is relevant for the given information or should it be ignored

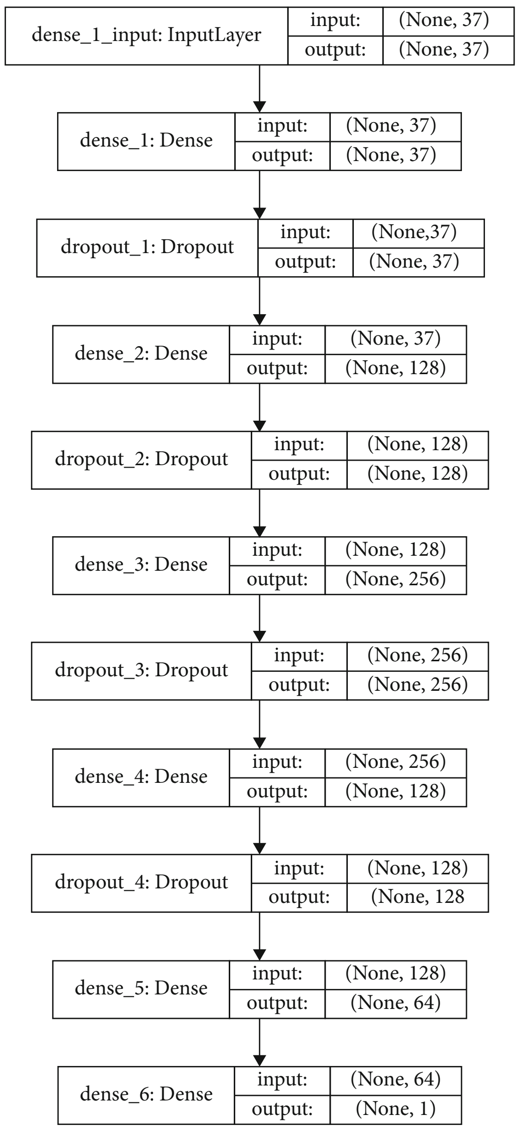

Figure 15: Best MLP architecture obtained for our framework (obtained from the Keras model visualization tool).

is calculated using this formula:

$$Y = \text{Activation}\left(\sum(\text{weight} \times \text{input}) + \text{bias}\right) \quad (3)$$

where weight represents the weights with which the layers are initialized. The input represents the input value for this layer. This input may be achieved from the input of the debut vector or values that come from the previous layer. Bias values permit to move the activation function to one side or right, which might be fundamental for successful learning. Many activation functions can be used; however, the system proposed in the present study involved the usage of the "RELU" function in hidden layers and "Sigmoid" in the last layer, to output a probabilistic result (to define the threshold of confidence as 85%) (Figure 14). The "Sigmoid" function provides a probabilistic result to the next layer, using an exponential calculation to choose which neurons

Table 4: Hyperparameters to be fine-tuned for both machine and deep learning architectures in order to get the best performance using Scikit-learn and Keras.

	Models					
	SVM		KNN		CART	
	Parameter	Chosen values	Parameter	Chosen values	Parameter	Chosen values
Machine learning algorithms	C (penalty parameter C of the error term)	0.001, 0.01, 0.1, 1, and 10	K (number of neighbors)	From 1 to 30 with a step of 1	Criterion (measure the quality of a split)	Gini, entropy
			Weight (how weights are initialized)	Uniform, distance	Splitter (method of the split at each node)	Best, random
			Leaf size (brute force searches between nodes)	50, 100, 200, 300, and 400	Number of features (number of features to consider when looking for the best split)	Auto, sqrt, log2
	Gammas (how far the influence of a single training example reaches)	0.001, 0.01, 0.1, and 1	Algorithm (how it will look for neighbors)	Auto, ball_tree, kd_tree, brute	Minimum of samples split (minimum number of samples required to split an internal node)	2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15
			P (power parameter for the Minkowski metric, defines how to calculate the distance)	1, 2, and 3	Minimum of leaf samples (minimum number of samples required to be at a leaf node)	2, 3, 4, 5, 6, 7, 8, 9, and 10
			Metric (the distance metric to use for nodes of the tree)	Minkowski, Euclidean, Manhattan, and Chebyshev	Maximum depth (maximum depth of the tree)	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15
	Kernel (pattern analysis method)	RBF, linear			Maximum features (No. of features to consider when looking for the best split)	From 1 to 36 with a step of 1 in each increasing
	Architectures					
	MLP		CNN			
	Parameter	Chosen values	Parameter	Chosen values		
Deep learning architectures	Units (number of neurons in layer)	32, 64, 128, 256, and 512	Filter number (matrix filer for convolutional calculation)	5, 10, 15, 20, 25, and 30		
	Batch size (how many samples to treat before adjusting weights)	32, 64, 96, 128, and 256	Filter length	1, 2, 3, 4, 5, and 6		
	Epochs (iterations for training model)	50, 100, 250, 500, and 1000				

can be used or not, and this is through normalizing the value of weights used. On the other hand, the "RELU" function returns the maximum between zero and the value of weights.

This type of architectures are further capable to define a number of hidden layers and each layer with different number of neurons, and for each layer, a different manner can be selected to initialize weights to be injected into the activation function using "Regularizes," such as L2 or L1, and from a layer to another different techniques can be used to reduce the number of neurons that do not work as well as expected by using the dropout function that takes in the parameter a percentage of neurons that can be ignored if they do not give the expected result for reducing the loss function or when there is no proven accuracy in order to avoid the overfitting phenomenon. This can help to learn from data by steps and with fine-tuning hyperparameters. However, this takes much time in comparison with different chosen architectures and with classic machine learning systems.

In developing a deep learning architecture, several things are considerable for instance: the number of layers used in the system, the number of neurons used in every research, etc. Certain propositions have been made by the researchers regarding the development of DLNN (deep learning neuronal network). Some of them includes the following:

(1) "A rule of thumb is for the size of this [hidden] layer to be somewhere between the input layer size…and the output layer size…" [56]

Table 5: Best hyperparameters obtained for each model.

	Algorithm	Best model got

Machine learning	KNN	algorithm="auto"
		leaf_size = 400
		metric="Manhattan"
		metric_params=none
		n_jobs = 1
		n_neighbors = 6
		$p = 2$
		weights="distance"

	CART	criterion="entropy"
		max_depth = 12
		max_features = 25
		min_samples_leaf = 4
		min_samples_split = 13
		presort=true
		random_state = 123
		splitter="best"

	SVM	C = 10.0
		Gamma = 0.1
		Kernel="rbf"

Deep learning	Architecture	Best model got
	MLP	Shown in Figure 14
	CNN	Shown in Figure 15

(2) "To calculate the number of hidden nodes, a general rule of $(\text{Number of inputs} + \text{outputs}) \times (2/3)$" has been selected (from the FAQ for a commercial neural network software company)

(3) "You will never require more than twice the number of hidden units as you have inputs" in an MLP with one hidden layer [57]

(4) "How large should the hidden layer be? One rule of thumb is that it should never be more than twice as large as the input layer" [58]

(5) "Typically, unlimited hidden nodes were specified as dimensions [principal components] needed to capture 70-90% of the variance of the input data set." [59]

The opinions and architectures used are varied leading towards the idea that there is no exact way to know how to build networks. Still, one of the major advantages of incorporating the deep learning is that it is easy to build networks to work with good accuracy and less error rate. The study further began with the multilayer perceptron (MLP) first-layer neurons by the number of input features which is 37 leading towards the addition of more hidden layers. Figure 15 provides the best MLP architecture that has been included in the framework of the present study. The overall process began with the first layer by 37 neurons as the input of our vector data uses L2 as a Regularizer for layer parameters, and it measures the weights and layer activity during the optimization. Those penalties are incorporated in the loss function that the network optimizes. After each layer, a dropout layer was incorporated to avoid overfitting. Hidden layers were built with 128, 256, 128, and 64 neurons, and all of them use "RELU" as activation function, and the last one with a single neuron because it will output one probabilistic value that will subsequently determine if it represents a phishing or benign result. Finally, the network should be compiled by using a loss function, as it is responsible in defining what the metric to use. This is useful in evaluating the training steps in the hidden layer. In order to do so, "binary_crossentropy" was selected from the Keras package which works as "penalty" score to reduce when training an algorithm on data. Also, "accuracy" as the performance metric measurement was chosen. The batch size consisted 128 samples, and epochs were set to 500 iterations.

Table 6: Confusion matrix.

		Actual class	
		No	Yes
Observed class	No	True negative	False positive
	Yes	False negative	True positive

For the convolutional neural network (CNN) architecture of the study framework, two layers of convolutional calculation were undertaken. The first one has 15 filters, and each filter was initialized with a matrix $3 * 3$ as feature map, and the second one has 15 filters, each one with a matrix 1×1 as feature map. This was then followed by a batch normalization layer which helps to standardize the data and accelerate the treatment. A max-pooling layer with a matrix of 2×2 max-pooling layer was incorporated which is a form of nonlinear downsampling. Max-pooling parcels the inputs into an arrangement of noncovering square shapes and, for each sublocale, yields the most extreme esteem. It helps to eliminate nonmaximal values, it reduces computation for the upper layers, and it provides a form of translation invariance. Then, last but not least, a flattening layer was used which transformed the output of the convolutional layers to a single vector which is used as a test vector for the final classification. The main goal to put a convolution calculation before an MLP network is that CNN will have solved the signal-translation and error rate problem, because they would convolve each input signal with a detector (kernel) and thus will be sensitive to the same feature. Finally, the performance of the three algorithms which include SVM, CART, KNN for machine learning and MLP and CNN for deep learning architectures was also compared to propose valuable findings.

Scikit-learn offers the possibility of a pipeline to put a collection of treatments to be executed in ordered steps, where first data were standardized with the StandardScaler package. Classifiers such as KNN, CART, or SVM were then added. The GridSearchCV package from Scikit-learn was then used to set a list of parameters as mentioned in Table 4 to be tested in combination. All these treatments were made in parallel tasks using the CPU only. The overall process was time-consuming as it took around 2 h for the machine learning algorithms but about 4 h for building deep learning models architectures.

Table 7: Considerable measurements to measure the performance of each model.

Measurement	Mathematical expression
Precision	TP/(TP + FP)
Recall (the recall measures the ability to find all true positive samples)	TP/(TP + FN)
Accuracy (all correct prediction and the overall tests made)	TP + TN/(TP + TN + FN + FP)
$f1$-score (weighted average of the precision and recall)	$2 \times$ (Precision $\times$ Recall)/(Precision + Recall)
Error rate	1-accuracy

Table 8: Results obtained after selecting the best models for each model.

		Measurements by %		
		Accuracy	$f1$-score	Error rate
Machine learning	SVM	91.132	92.556	9.8
	KNN	92.189	93.723	7.9
	CART	92.915	94.172	7.1
		Accuracy	$f1$-score	Error rate
Deep learning	MLP	93.216	94.752	6.8
	CNN	97.945	98.591	2.1

17. Results and Discussion

The provided phishing detection solution is effective as it analyses a URL in stress and from different points of view, against other solutions that are based on limited perspectives, i.e., usually 2 or 3 based methods that were discussed above. The overall formation of the solution was based on the process of extracting features which is very important during any model building and training, so without trusted features, it is not probable make a good prediction. This involves the system to understand the phenomenon better. At the end of this step, a binary vector was fitted to a learning algorithm. The classifier should contain correct answers to what the problem is about, so in the classification process, it will try to find which class belongs to the current element in the test. Findings of the study indicated positive results regarding the validity of the model. The system when tested through different techniques proposed positive results. For instance, deep learning was used to test the effectiveness of the classifiers in cases where data was increased. In this case, deep learning has shown outstanding performance.

Findings of the study further indicated the inclusion of MLP method is useful, since the architecture gave an accuracy of around 95.5% and loss score of 0.3% which is very satisfactory. Other findings are related to the usefulness of CNN which resulted in the reduction of error rate of 0.2%, leading towards the increased accuracy of 97.945%.

Other findings are related to the comparison between the performance of algorithms namely SVM, CART, and KNN for machine learning and MLP and CNN for deep learning architectures which indicate that deep learning networks perform very well when data was increased and especially in cases where the number of hidden layers and neurons was also increased to a certain threshold. However, the utilization of CNN networks resulted in the accuracy of 97.945%.

Deep learning networks are further useful in providing an epoch and batch size. The study provided a list of parameters to be tested in order to have a good performance. Since there is no magical formula to determine the optimal network to which data depends on, it is only possible through hand experimentation. The activation functions such as "RELU" and "Sigmoid," number of epoch and batch size, and, for machine learning algorithms, a list of hyperparameters for each algorithm are provided. The table below represents a summary of the selected parameters. Note that fine-tuning hyperparameters are used as described in Scikit-learn and Keras libraries.

Table 5 then provides findings related to the best model attained for each test conducted through the process of Scikit-learn.

To measure the overall performance with the best built model, measures that were extracted from a confusion matrix were defined, as indicated in Table 6.

Findings of the study were provided in the following manner where a positive case is to detect a URL as phishing, and negative as benign URL.

So, the whole cases can be defined as follows:

(1) TP (true positive): phishing URLs that were correctly classified as phishing

(2) TN (true negative): benign URLs that were correctly classified as benign

(3) FN (false negative): phishing URLs that were classified as benign

(4) FP (false positive): benign URLs that were classified as phishing. Moreover, the measures are defined in Table 7.

Table 8 then provide results regarding the use of dataset which contained around 40000 URLs. According to the findings, in machine learning, the 91.132% accuracy was attained for SVM, 92.189% for KNN, and 92.915% for CART, whereas for deep learning, the highest accuracy score 97.945% was attained for CNN while 93.216% for MLP.

As evident from the table, the CNN architecture demonstrated the best results due to its capability to reduce error rate in each step of the convolutional calculation.

When comparing different methods, findings indicated that several solutions used two to three different perspective to analyse a URL, whereas the method involved in the present study uses six different algorithm based-methods. To do

Table 9: Comparative analysis between solutions proposed in the present study and other studies.

Solution	Based perspective	Accuracy	Error rate	Advantage	Disadvantage
Cantina [60]	Content and identity (static analysis)	95%	5%	Fast Can be integrated with phishing toolbars	Information gathered is still reduced Knowledge about URL is limited
Daeef et al. [61]	Lexical and machine learning (hybrid analysis)	92.24%	5.40%	Wide scope and fast phishing detection system	High false positive rate
Yang, Zhao, Zen. [62]	Blacklist, lexical, and deep learning CNN (static analysis)	98.99%	0.59%	Fast Based on deep learning	Needs improvement and more features
Jain and Gupta [63]	Visual similarity and machine learning (static analysis)	99.72%	1.89%	Fast to recognize targeted victims	Limited to e-banking websites Knowledge about URL is limited
Solution provided in the existing study	Blacklist Lexical Content Identity Visual similarity Behavioral Machine or deep learning (hybrid analysis)	97.94%	2.1%	Fast Based on rules Trusted Complete knowledge about a URL	Time and resource consuming when the whole process should be performed

so, the accuracy and the error rate were compared with the solution provided in this study. For example, a solution that uses only blacklist and lexical will not have information about other perspectives like the content, visual, or behavioral of the URL in question. This indicates that the solution provided in the present study is complete and accurate depending upon the functionality of the solutions provided in other studies. However, similar results have been used in the study to propose valid and valuable findings (Table 9).

The overall discussion was based on the value addition of ten layers (1-10) which (1) includes the use of lengthy URL and the incorporation of IP address rather than the DNS name, (2) includes the increasing number of dots within the address and the usage of modified port number, (3) is related to the use of suspicious SSL certificate along with domain age, (4) was related to the unsecured and longer time for account accessibility, (5) was related to the use of Java scripts to hide information, (6) includes the visual similarity of other pages along with black listed URL name, (7) is associated to the usability of forms with submit buttons and pop-up windows, (8) refers to the importance on security and response, (9) was related to the use of mouse over for link obscure, and (10) is related to the use of prefixes and suffices in web address bar and hexadecimal characters and symbols. Findings of the study indicated that the following rules were helpful in the detection of phishing. It further indicated that the absence or presence of some of the characteristics provided high probability for the presence of phishing websites. Finally, as per the findings, the study concluded that any URL is said to be phishing if it obeys most of the selected rules. These findings are in line with the findings proposed in the present study, where following rules were incorporated in the form of features.

Content analysis was performed to identify the key phrases used by the attackers. Findings of the study indicated that the content analysis was effective in identifying the phishing techniques used through different sources to gain individual's personal information illegitimately. Another study was conducted by Riaty et al. [62] who focused on the methods to enhance the detection of phishing websites. Central focus was granted to the machine learning techniques associated to fuzzy logic and rules. Findings of the study indicated the effectiveness of clustering, frequent pattern mining, and value mapping process. It was further recommended to incorporate the preprocessing and fuzzy system prior to phishing detection to enhance the accuracy of identifying the phishing websites. These findings are in contrast with those proposed in the present study, as deep learning was more effective in comparison to the machine learning.

Despite the presence of abundant knowledge, the study involved certain limitations. Major limitation of the study is utilization of only six algorithm methods. Secondly, though the proposed method has high accuracy and validity, the long-time duration of the provided may limit the practical application of the given method.

18. Conclusion

This paper presents a new hybrid solution to detect and prevent from phishing URLs. The novelty in our solution is that it is based on reasonable rules to improve the logic of treatments as in certain cases it is probable to detect a URL without executing the whole process. However, considering the applicability, accuracy, and reliability of the method, the study suggested to undertake a complete process to update the model and extract new knowledge about how phishing phenomenon changes. It helps us to use dynamic features extraction. The study reflects that by doing so, it is possible to achieve valuable results in classification using only a part of all features. The study further presented the classification of features using machine learning and deep learning models with their hyperparameter fine-tuning process.

Deep learning networks have shown an enormous capability to resolve the problem of training from data especially with huge data. It is usually used for computer vision as image detection and classification problems. However, with regards to the machine learning field, researches are based on good concept algorithms. Still, trainings associated to deep learning networks need more resources due to the huge mathematical calculations that can be made, and in the case of binary classification for phishing URL detection, deep learning models especially CNN have not only demonstrated a very good performance and accuracy but also helped in the reduction of error rate. Contributions of the present study are valuable as it has developed a unique method by the amalgamation of six different algorithm methods. To the best of researcher's knowledge, such a study has not been conducted before. Since the study incorporated 37 different features, this develops that a stress test on URLs conducted from a different approach of analysis is good enough rather than using a single mode of analysis such as blacklist or lexical. Also, the system introduced in the present study used static analysis at first rather than dynamic to remain independent from software or hardware that are generally needed for the execution of web page in question.

As a future work, it is recommended to scale this solution to be used in production. However, depending upon the study results and above discussion, it can be inferred that the solution proposed in this study serve as a trusted solution for batch and offline use. Future researchers are suggested to expand the area of knowledge by focusing on the following solutions:

(i) What is the efficient minimal set of features that can be used to predict a phishing URL?

(ii) How to take advantage of big data (volume, veracity, and velocity), and how to integrate it into deep learning models?

(iii) How to extend the project to support other types of attacks: spam and malware URLs?

(iv) Opportunities to perform better-preventing attacks before they can happen and to mitigate zero-hour attacks.

Data Availability

Data available upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This project was funded by the Deanship of Scientific Research at King Abdulaziz University, Jeddah, under grant No. RG-7-611-39. The authors, therefore, acknowledge with thanks DSR for technical and financial support.

References

[1] L. James, "Chapter 1: banking on phishing," in *Phishing Exposed*, L. James, Ed., Syngress, Rockland, 2006.

[2] P. McFedries, "Technically speaking: gone phishin," *IEEE Spectrum*, vol. 43, no. 4, p. 80, 2006.

[3] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: a literature survey," *IEEE Communications Surveys & Tutorials*, vol. 15, pp. 2091–2121, 2013.

[4] S. Purkait, "Phishing countermeasures and their effectiveness – literature review," *Information Management & Computer Security*, vol. 20, pp. 382–420, 2012.

[5] E. E. Lastdrager, "Achieving a consensual definition of phishing based on a systematic review of the literature," *Crime Science*, vol. 3, no. 1, 2014.

[6] G. Ollmann, "The phishing guide–understanding & preventing phishing attacks," *NGS Software Insight Security Research*, 2004.

[7] Z. Ramzan and C. Wueest, "Phishing attacks: analyzing trends in 2006," *CEAS*, 2007.

[8] A. Litan, "Increased phishing and online attacks cause a dip in consumer confidence," *Gartner Study (June 2005)*, 2005.

[9] L. L. Sullins, "Phishing for a solution: domestic and international approaches to decreasing online identity theft," *Emory Int'l L. Rev*, vol. 20, 2006.

[10] Cyren, "Cyber Threats Report the Growing Risk to Business Data Q1," *Editor: Book cyber threats report the growing risk to business data Q1*, 2015.

[11] APWG, "APWG Trends Report q1 2018," *Editor: Book APWG trends report q1 2018*, 2018.

[12] Symantec, "Internet Security Threat Report," *Editor: Book Internet Security Threat Report*, 2016.

[13] Kaspersky, *Financial Cyber Threats In 2017*, 2018.

[14] J. McCabe, *FBI Warns of Dramatic Increase in Business E-Mail Scams*, Phoenix, 2016, November 2018, https://www.fbi.gov/contact-us/field-offices/phoenix/news/press-releases/fbi-warns-of-dramatic-increase-in-business-e-mail-scams.

[15] APAC, "Monthly Phishing Website Processing Briefing," *Editor: Book Monthly phishing website processing briefing*, 2017.

[16] Google Safe Browsing, *Safety First*December 2018, https://safebrowsing.google.com/.

[17] N. C. R. L. Y. Teraguchi and J. C. Mitchell, *Client-Side Defense against Web-Based Identity Theft*, University, 2004, http://crypto.stanford.edu/SpoofGuard/webspoof.

[18] G. Varshney, M. Misra, and P. K. Atrey, "A survey and classification of web phishing detection schemes," *Security and Communication Networks*, vol. 9, 6284 pages, 2016.

[19] A. Jain and B. B. Gupta, "Phishing detection: analysis of visual similarity based approaches," *Security and Communication Networks*, vol. 2017, 20 pages, 2017.

[20] M. Vazirgiannis, D. Drosos, P. Senellart, and A. Vlachou, "Web page rank prediction with Markov models," in *Proceedings of the 17th international conference on World Wide Web*, Beijing, China, 2008.

[21] L. J. Singh and N. I. Imphal, “A survey on phishing and anti-phishing techniques,” *International Journal of Computer Science Trends and Technology (IJCST)*, vol. 6, pp. 62–68, 2018.

[22] S. Marchal, *DNS and Semantic Analysis for Phishing Detection*, Doctoral dissertation, Université de Lorraine, 2015.

[23] C. Iuga, J. R. Nurse, and A. Erola, “Baiting the hook: factors impacting susceptibility to phishing attacks,” *Human-centric Computing and Information Sciences*, vol. 6, p. 8, 2016.

[24] M. Aburrous, M. A. Hossain, K. Dahal, and F. Thabtah, “Intelligent phishing detection system for e-banking using fuzzy data mining,” *Expert Systems with Applications*, vol. 37, pp. 7913–7921, 2010.

[25] C. I. Canfield, B. Fischhoff, and A. Davis, “Quantifying phishing susceptibility for detection and behavior decisions,” *Human Factors*, vol. 58, pp. 1158–1172, 2016.

[26] X. Dong, J. A. Clark, and J. L. Jacob, “User behaviour based phishing websites detection,” in *2008 International Multiconference on Computer Science and Information Technology IEEE*, pp. 783–790, Wisla, Poland, 2008.

[27] PhishTank, *Join the fight against phishing*, 2018, December 2018, https://www.phishtank.com/.

[28] Alexia, *Find, Reach, and Convert Your Audience*, 2018, December 2018, https://www.alexa.com/.

[29] RFC 1738, 2019, https://www.ietf.org/rfc/rfc1738.txt.

[30] Virus Total, *Getting Started*December 2018, https://developers.virustotal.com/reference#getting-started.

[31] P. Likarish, E. Jung, and I. Jo, “Obfuscated malicious Java Script detection using classification techniques,” in *2009 4th International Conference on Malicious and Unwanted Software (MALWARE)*, Montreal, QC, Canada, 2009.

[32] J. S. White, J. N. Matthews, and J. L. Stacy, “A Method for the Automated Detection Phishing Websites through Both Site Characteristics and Image Analysis,” in *Cyber Sensing 2012*, no. article 84080, 2012International Society for Optics and Photonics, 2012.

[33] Y. T. Hou, Y. Chang, T. Chen, C. S. Laih, and C. M. Chen, “Malicious web content detection by machine learning,” *Expert Systems with Applications*, vol. 37, pp. 55–60, 2010.

[34] C. Soman, H. Pathak, V. Shah, A. Padhye, and A. Inamdar, “An intelligent system for phish detection, using dynamic analysis and template matching,” *World Academy of Science, Engineering and Technology*, vol. 42, pp. 321–327, 2008.

[35] F. Thabtah and F. Kamalov, “Phishing detection: a case analysis on classifiers with rules using machine learning,” *Journal of Knowledge Management*, vol. 16, article 1750034, 2017.

[36] S. Smadi, N. Aslam, and L. Zhang, “Detection of online phishing email using dynamic evolving neural network based on reinforcement learning,” *Decision Support Systems*, vol. 107, pp. 88–102, 2018.

[37] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[38] J. Shlens, *A Tutorial on Principal Component Analysis*, 2014.

[39] S. Nembrini, *Machine Learning Methods for Feature Selection and Rule Extraction in Genome-Wide Association Studies (GWASs)*, 2013.

[40] K. A. Norman, S. M. Polyn, G. J. Detre, and J. V. Haxby, “Beyond mind-reading: multi-voxel pattern analysis of fMRI data,” *Trends in Cognitive Sciences*, vol. 10, pp. 424–430, 2006.

[41] T. Hayes, S. Usami, R. Jacobucci, and J. J. McArdle, “Using classification and regression trees (CART) and random forests to analyze attrition: results from two simulations,” *Psychology and Aging*, vol. 30, p. 911, 2015.

[42] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, “KNN model-based approach in classification,” in *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, Berlin, 2003.

[43] C. W. Hsu, C. C. Chang, and C. J. Lin, *A Practical Guide to Support Vector Classification*, 2003.

[44] D. Sahoo, C. Liu, and S. C. Hoi, *Malicious URL Detection Using Machine Learning: A Survey*, 2017.

[45] R. Basnet, S. Mukkamala, and A. H. Sung, “Detection of Phishing Attacks: a Machine Learning Approach,” in *Soft Computing Applications in Industry*, Springer, Berlin, 2008.

[46] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Learning to detect malicious URLs,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, p. 30, 2011.

[47] Y. Mourtaji, M. Bouhorma, and D. Alghazzawi, “New phishing hybrid detection framework,” *Journal of Theoretical & Applied Information Technology*, vol. 96, 2018.

[48] F. D. Abdi and L. Wenjuan, “Malicious URL detection using convolutional neural network,” *Journal International Journal of Computer Science, Engineering and Information Technology*, vol. 7, pp. 1–8, 2017.

[49] R. Hassanpour, E. Dogdu, R. Choupani, O. Goker, and N. Nazli, “Phishing E-mail detection by using deep learning algorithms,” in *Proceedings of the ACMSE 2018 Conference ACM*, vol. 45, Richmond, KY, USA, 2018.

[50] A. J. Shepherd, *Second-Order Methods for Neural Networks: Fast and Reliable Training Methods for Multi-Layer Perceptrons*, Springer Science & Business Media, 2012.

[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, 2012.

[52] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[53] Scikit Learn, *Machine Learning in Python*, 2018, December 2018, https://scikit-learn.org/stable/.

[54] Keras, *Keras: The Python Deep Learning library*, 2018, December 2018, https://keras.io/.

[55] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, *Comparative Study of Deep Learning Software Frameworks*, 2015.

[56] A. Blum, *Neural Networks in C++: an Object-Oriented Framework for Building Connectionist Systems*, John Wiley & Sons, Inc, New York, 1992.

[57] K. Swingler, *Applying Neural Networks: a Practical Guide*, Morgan Kaufmann, Academic Press, London, 1996.

[58] M. J. Berry and G. S. Linoff, *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management*, John Wiley & Sons, New York, 2004.

[59] Z. Boger and H. Guterman, “Knowledge extraction from artificial neural network models,” in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, pp. 3030–3035, Orlando, FL, USA, 1997.

[60] Y. Zhang, J. I. Hong, and L. F. Cranor, “Cantina: a content-based approach to detecting phishing web sites,” in *Proceedings of the 16th international conference on World Wide Web*, pp. 639–648, Banff, Alberta, Canada, 2007.

[61] A. Y. Daeef, R. B. Ahmad, Y. Yacob, and N. Y. Phing, “Wide scope and fast websites phishing detection using URLs lexical features,” in *2016 3rd International Conference on Electronic Design (ICED)*, pp. 410–415, Phuket, Thailand, 2016.

[62] P. Yang, G. Zhao, and P. Zeng, “Phishing website detection based on multidimensional features driven by deep learning,” *IEEE Access*, vol. 7, pp. 15196–15209, 2019.

[63] A. K. Jain and B. B. Gupta, “Detection of phishing attacks in financial and e-banking websites using link and visual similarity relation,” *International Journal of Information and Computer Security (IJICS)*, vol. 10, pp. 398–417, 2018.