IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 10, 2022, accepted April 11, 2022, date of publication April 18, 2022, date of current version April 27, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3168681

Phishing URL Detection: A Real-Case Scenario Through Login URLs

MANUEL SÁNCHEZ-PANIAGUA, EDUARDO FIDALGO FERNÁNDEZ, ENRIQUE ALEGRE, WESAM AL-NABKI, AND VÍCTOR GONZÁLEZ-CASTRO

Department of Electrical, Systems and Automation Engineering, Universidad de León, 24071 León, Spain
INCIBE—Spanish National Institute of Cybersecurity, 24005 León, Spain

Corresponding author: Manuel Sánchez-Paniagua (msancp@unileon.es)

This work was supported by the framework agreement between the University of León and INCIBE (Spanish National Cybersecurity Institute) under Addendum 01.

ABSTRACT Phishing is a social engineering cyberattack where criminals deceive users to obtain their credentials through a login form that submits the data to a malicious server. In this paper, we compare machine learning and deep learning techniques to present a method capable of detecting phishing websites through URL analysis. In most current state-of-the-art solutions dealing with phishing detection, the legitimate class is made up of homepages without including login forms. On the contrary, we use URLs from the login page in both classes because we consider it is much more representative of a real case scenario and we demonstrate that existing techniques obtain a high false-positive rate when tested with URLs from legitimate login pages. Additionally, we use datasets from different years to show how models decrease their accuracy over time by training a base model with old datasets and testing it with recent URLs. Also, we perform a frequency analysis over current phishing domains to identify different techniques carried out by phishers in their campaigns. To prove these statements, we have created a new dataset named Phishing Index Login URL (PILU-90K), which is composed of 60K legitimate URLs, including index and login websites, and 30K phishing URLs. Finally, we present a Logistic Regression model which, combined with Term Frequency - Inverse Document Frequency (TF-IDF) feature extraction, obtains 96.50% accuracy on the introduced login URL dataset.

INDEX TERMS Cybercrime, login, machine learning, phishing detection, URL.

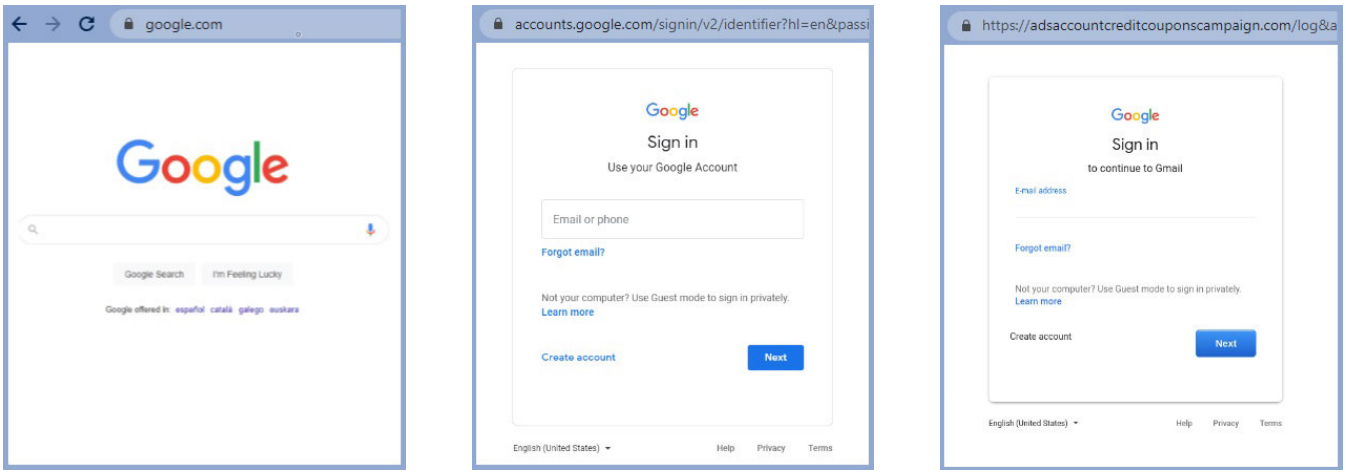
I. INTRODUCTION

In the last years, web services usage has grown drastically due to the current digital transformation. Companies motivate the change by providing their services online, like e-banking, e-commerce or SaaS (Software as a Service) [1]. Nowadays, due to the COVID-19 pandemic, restrictions have spread out the work-from-home model, which implies extra millions of workers, students, and teachers developing their activities remotely [2], leading to a substantial additional workload for services such as email, student platforms, VPNs or company portals. Therefore, there are even more potential targets exposed to phishing attacks, where phishers try to mimic legitimate websites to steal users' credentials or payment information [3], [4]. Recent studies [5], [6] concluded that phishing is one of the most significant attacks based on social engineering during the COVID-19 pandemic, together with spam emails and websites to execute these attacks.

Identifying phishing sites through their HTTP protocol is no longer a valid rule. In the 3^{rd} quarter of 2017 [7], the APWG reported that less than 25% of phishing websites were hosted under HTTPS protocol, whilst this amount has increased up to 83% in 1^{st} quarter of 2021 [8]. These websites provide secure end-to-end communication, which transmits a false safe impression to the user while making an online transaction [9]. Furthermore, the Anti-Phishing Working Group (APWG) [10] has reported a significant increase in phishing attacks, i.e. from 165, 772 to 611, 877 websites, just between the first quarter of 2020 and 2021 respectively. A reason behind this increase might be that people have resorted (and still are) to online services during the COVID-19 pandemic.

One of the most popular solutions for phishing detection is the list-based approach, which analyzes the requested

The associate editor coordinating the review of this manuscript and approving it for publication was Senthil Kumar.



(a) Legitimate homepage (b) Legitimate login page (c) Phishing web page

FIGURE 1. Difference between legitimate home (a), legitimate login (b) and phishing (c) pages. Samples like (a) are commonly used in state-of-the-art approaches. We introduce in our dataset samples like (b), which has a similar look to phishing attacks like (c).

URL against a phishing database [11]. Some examples of this solution are Google SafeBrowsing,[1] PhishTank,[2] OpenPhish[3] or SmartScreen.[4] If a requested URL matches any record, the request is blocked, and a warning is displayed to the user before visiting the website. However, despite the capabilities of the list-based approach, it would fail if the phishing URL was not reported previously [12]–[14], and it will require a continuous effort to update the database with newer phishing data. Bell and Komisarczuk [11] observed that many phishing URLs were removed after day five from Phishtank while OpenPhish removed all URLs after seven days from its report. This issue allows attackers to reuse the same URL when it is removed from different lists.

Due to the mentioned drawbacks with the blacklist-based methods, automatic detection of phishing URLs based on machine learning, have attracted attention in research [15], [16]. These approaches can be grouped into four classes according to the type of data used for the detection: the text of the URL, the page content, the visual features and networking information [17]. Methods based on the page content and visual features require visiting the website to collect the source code and render it, which is a time-consuming task. Other availability limitations can be found in studies that rely on networking and 3^{rd} party information such as WHOIS or search engine rankings. To overcome these limitations, we focus on phishing detection through URLs since it implies advantages such as fast computation -because no websites are loaded- and 3^{rd} party and language independent, since features are extracted only from the URLs.

Existing URL datasets use the homepage URL from well-known websites as the legitimate [18], [19]. However, we think that the challenge is to determine if a *login form* of a website is legitimate or phishing. From our perspective, and to the best of our knowledge, publicly available datasets are not reflecting conditions that represent some real problems for phishing URL detection. Fig. 1 displays the differences between a homepage, a login page and a phishing website. Furthermore, it is observed that recent machine learning proposals obtained high accuracy using outdated datasets, i.e., typically containing URLs collected from 2009 to 2017. We demonstrate that models trained with old URLs decrease their performance when they are tested with URLs coming from recent phishing pages.

This paper presents a phishing URL dataset using legitimate login websites to obtain the URLs from such pages. Then, we evaluate machine and deep learning techniques for recommending the method with higher accuracy. Next, we show how models trained with legitimate homepages struggle to classify legitimate login URLs, demonstrating our hypothesis about phishing detection and legitimate login URLs. Additionally, we show how the accuracy decrease with the time on models trained with datasets from 2016 and evaluated on data collected in 2020. Finally, we provide an overview of current phishing encounters, explaining attacker tricks and approaches.

The main contributions of the paper can be summarized as follows:

- We extended our previous dataset PILU-60K (Phishing Index Login URL) [20], from 60K to 90K URLs equally distributed among three classes: phishing, the legitimate homepage, and legitimate login. We make this extended dataset, PILU-90K, publicly available for research purposes[5]
- Using PILU-90K, we implemented and evaluated three pipelines for URL phishing detection: (i) we

[1]https://safebrowsing.google.com/
[2]https://www.phishtank.com/
[3]https://openphish.com/
[4]https://bit.ly/2OJDYBS
[5]https://gvis.unileon.es/dataset/pilu-90k/

use the 38 handcrafted feature descriptors proposed by Sahingoz *et al.* [21] for training eight supervised machine learning classifiers and also (ii) automatic feature extraction using Term Frequency Inverse Document Frequency (TF-IDF) at character N-gram level combined with Logistic Regression (LR) algorithm, and (iii) a Convolutional Neural Network (CNN) at character level too.
- We demonstrated empirically how an URL phishing detection model struggles in classifying login URLs when it was trained on the URLs of the homepage of phishing and legitimate URLs.
- We evaluated the robustness of the proposed phishing detection over time. We trained the model on a dataset collected between March 2016 and April 2016, and we evaluated the model on other datasets collected between 2017 and 2020.
- Phishing websites were analyzed using domain frequency. We found six different phishing domains depending on the service hired by the attacker.

The organization of the paper is as follows: Section II reviews the literature on phishing detection. Next, Section III describes the proposed dataset and its content. Then, we explain the used features and the proposed classifiers in Section IV. The carried out experiments are covered in Section V. Section VI presents and discusses the obtained results. Finally, the main conclusions are drawn in Section VII, where we also point to our future work.

II. STATE OF THE ART

In the literature, researchers have focused on phishing detection following three main approaches: *List-based* and automatic detection using *Machine Learning* and *Deep Learning* techniques.

A. LIST-BASED

The list-based approach, well-known for detecting phishing URLs [22]–[24], can be based on whitelists or blacklists, depending if they store legitimate or phishing URLs, respectively. Jain and Gupta [24] developed a whitelist-based system that blocks all websites which are not on that list. Conversely, the blacklist-based systems, like Google Safe Browse or PhishNet [23], are more common as they provide a zero false-positive rate, i.e. no legitimate website is classified as phishing. However, they can be compromised if an attacker makes changes on a blacklisted URL. Besides, they depend heavily on the update rate of the system's records. Therefore, a list-based approach is not a robust solution due to the high volume of new phishing websites introduced daily and their short lifespan, which is estimated to be 21 days on average [12].

B. MACHINE LEARNING METHODS

To overcome blacklist disadvantages, researchers have developed machine learning models to detect unreported phishing encounters. Depending on their input data, these approaches can be classified into two categories: URL-based and content-based.

1) URL-BASED

Buber *et al.* [25] implemented a URL detection system composed of two sets of features. The first was a 209 word vector, obtained with "StringToWordVector" tool from Weka.[6] The second, 17 NLP (Natural Language Processing) handcrafted features such as the number of sub-domains, random words, digits, special characters and length measurements over the URL words. Combining both feature sets, they obtained a high 97.20% accuracy with Weka's RFC (Random Forest Classifier) on a 10% sub-sample set from Ebbu2017 dataset. In the following studies, Sahingoz *et al.* [21] defined three different feature sets: Word vectors, NLP and a hybrid set combining both sets. They obtained a 97.98% accuracy on Random Forest (RF) using only 38 NLP features on Ebbu2017 [25] dataset. In this work, we used the NLP features from Sahingoz *et al.* [21], since they reported state-of-the-art performance in the last studies.

Jain and Gupta [26] built an anti-phishing system using 14 handcrafted URL descriptors, including some obtained using 3^{rd} party services like WHOIS registers or DNS lookups. They obtained an accuracy of 76.87% and 91.28% with Naïve Bayes (NB) and Support Vector Machine (SVM) classifiers, respectively, on a private dataset with 35, 491 samples.

Banik and Sarma [27] implemented a lexical feature selection from URL to optimize the number of features and the accuracy of their model. They started with a set of 17 descriptors and removed the less significant ones until they reached an optimal performance. Using 9 features and a Random Forest (RF) classifier they obtained 98.57% accuracy on an extension of PWD2016 [18] dataset.

2) CONTENT-BASED

Content-based works use features extracted mainly from the websites' source code. However, most of the current works combine these with URLs and other 3^{rd} party services such as WHOIS [28], [29].

One of the first content-based works was CANTINA [30], which consists of a heuristic system based on TF-IDF. CANTINA extracts five words from each website using TF-IDF and introduced them into the Google search engine. If a domain was within the n first results, the page was considered legitimate, or phishing otherwise. They obtained an accuracy of 95% with a threshold of $n = 30$ Google search results. Due to the use of external services like WHOIS[7] and the high false-positive rate, authors proposed CANTINA+ [31]. Their new proposal achieved a 99.61% F1-Score including two filters: (i) a comparison of hashed HTML tags with known

[6]https://www.cs.waikato.ac.nz/ml/weka/
[7]https://www.whois.net/

phishing structures and (ii) the discarded websites with no form.

Moghimi and Vorjani [32] proposed a system independent from third services like Google Page Rank or WHOIS. They used two handcrafted feature sets, extracted from the URL and the Document Object Model (DOM) of the website. The first set has nine legacy features including a set of keywords, while the second has eight novel features which inform of whether the website's resources are loaded using SSL protocol or not. They used Levenshtein distance [33] to detect typo-squatting by comparing the website and resources URLs. These features were used to train an SVM classifier and obtained an accuracy of 98.65% on their banking websites dataset.

Adebowale *et al.* [34] created a browser extension to protect users by extracting features from the URL, the source code, the images, and features extracted using third-party services like WHOIS. Those features were introduced into an Adaptive Neuro-Fuzzy Inference System (ANFIS) and combined with the Scale-Invariant Feature Transform (SIFT) algorithm, obtaining an accuracy of 98.30% on Rami *et al.* [35] dataset.

Rao and Pais [28] developed a phishing website classifier using the URL, the hyperlinks on the HTML code and third-party services including the age of the domain and the page rank on Alexa. They reached 99.31% accuracy with a Random Forest classifier.

Yang *et al.* [36] proposed an Extreme Learning Machine (ELM) model and established three different groups of features: (i) Surface features, composed of 12 URL handcrafted and 4 Domain Name System (DNS) features related to the registration date and the DNS records for that domain; (ii) 28 Topological features that are related to the structure of the website and (iii) 12 deep features related to the text and image similarity. Combining these sets of features and the ELM classifier, they obtained 97.5% accuracy.

Sadique *et al.* [37] presented a framework for real-time phishing detection using four sets of URL features: (i) Lexical features related to the number of characters, dots and symbols found in different parts of the URL, (ii) host-based features that are related to the host, (iii) WHOIS features are related to the registration date and (iv) GeoIP-based features like the Autonomous System Number (ASN). A total of 142 individual features were evaluated using 98, 000 samples from Phishtank, where legitimate samples are also picked from false positives collected at PhishTank. They obtained a 90.51% accuracy on a Random Forest classifier using the proposed descriptors.

Li *et al.* [29] presented a stacking model which was the combination of three models: Gradient Boost Decision Tree (GBDT), eXtreme Gradient Boosting (XGBoost) and Light Gradient Boosting Model (LGBM). This stacking model was fed with a set of features from different sources: eight from the URL, 11 from the HTML and HTML string embeddings inspired by Word2Vec model [38]. They obtained 97.30% accuracy using a 49, 947 samples dataset.

C. DEEP LEARNING

Regarding the methods based on Deep learning, Somesha *et al.* [39] proposed a model based on Long Short-Term Memory (LSTM) to classify phishing URLs using ten handcrafted features from Rao and Pais [28]. Those features are three URL features based on the number of dots, the length of the URL, and the presence of HTTPS, six features extracted from the HTML, including the internal links and images, the ratio of broken links and the presence of anchor links on the HTML body. Finally, one third-party numeric feature was obtained from Alexa's Page Rank. These features were extracted from a 3, 526 samples dataset and introduced into the LSTM model to obtain 99.57% accuracy.

Aljofey *et al.* [40] presented an RCNN model to classify phishing URLs. They used the URL as input for a tokenizer and then used a one-hot encoding to represent the URL as a matrix at a character level. The last step is to set a fixed length of 200 characters for the model input. If the URL is under that threshold, the remaining characters are filled with zeros. Otherwise, the characters above the limit are trimmed. Finally, they used a 310, 642 URL dataset to feed an RCNN model, which obtained 95.02% using the aforementioned character embedding level features.

Al-Alyan and Al-Ahmadi [41] proposed a modified Convolutional Neural Network (CNN). First, they omitted the URL protocol and then cropped URLs larger than 256 characters. They used a 69 characters alphabet with lower-case letters, numbers and some symbols to obtain a 128 embedding vector. Then, a one-dimensional CNN was applied to obtain 95.78% accuracy on a 2, 307, 800 URLs dataset.

Zhao *et al.* [42] presented a Gated Recurrent Neural Network (GRU) capable of learning sequences and patterns within the URLs. They compared this approach against a set of 21 handcrafted features combined with an RF classifier. Results showed how automatic feature extraction combined with GRUs outperformed RF, reaching 98.5% and 96.4% respectively.

III. DATASET: PHISHING INDEX LOGIN URLs (PILU-90K)

Phishers use login forms to retrieve and steal users' data. As far as we are concerned, the legitimate class in most phishing datasets are represented by URLs from their homepages [18], [19]. However, most websites have their login form in different locations, making models trained with such public datasets to be biased since the URLs of homepages tend to be shorter and simpler than others. An example of this is depicted in Figure 2.

In this paper, we present an extended version of the Phishing Index Login URL (PILU-60K) dataset [20] and we name it PILU-90K. PILU-90K contains 90K URLs divided into three classes (see Figure 2): 30K legitimate URLs of homepages, 30K legitimate login URLs and 30K phishing URLs.

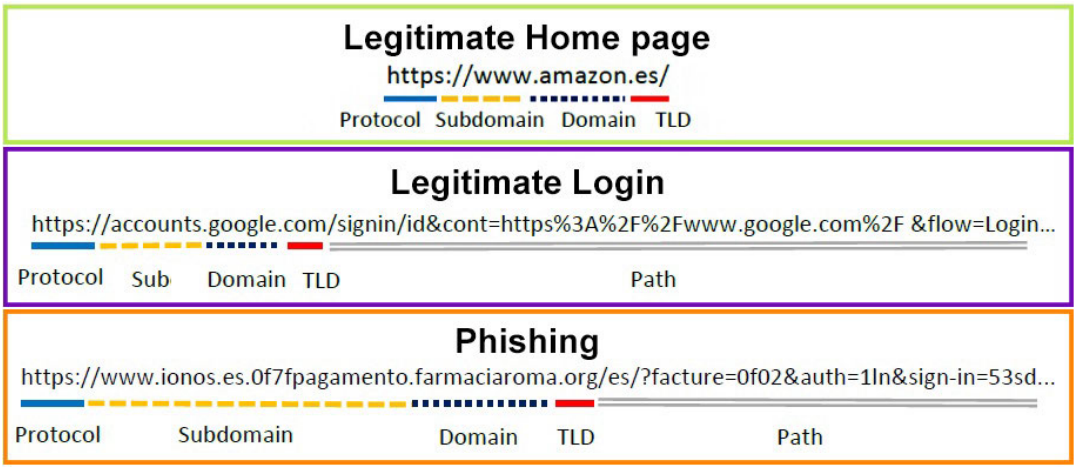


FIGURE 2. Types of URLs in PILU-90K and their parts. A homepage URL (up), a login page URL (middle) and a phishing URL (bottom). The length variation between a legitimate login page and a phishing one is minimum.

TABLE 1. Number of samples distributed in the different subsets used in this work.

Subset	Legit Index	Phishing	Legit Login
PIU-60K	30,000	30,000	-
PLU-60K	-	30,000	30,000

We collected the legitimate URLs from the Top Million Quantcast website,[8] which provides the most visited domains from the United States. The list provided on that website only contains the domain names, so we visited them to extract the complete URL. To reach the login page from a website, we used the Selenium web driver[9] and Python, checking buttons or links that could lead to the login form web page. Once we found the presumptive login, we inspected if the form had a password field in order to confirm whether it was a login form. Otherwise, it was not added to the dataset. We collected reported phishing URLs from Phishtank [21], [36], [39], between November 2019 and February 2020.

In this work, we have built two subsets from the PILU-90K dataset to conduct the proposed experiments. The first one, named PIU-60K (Phishing Index URLs), is built using the URLs of both the homepages of the legitimate samples and the phishing ones, following the configuration of most of the current state-of-the-art approaches. The second one, PLU-60K (Phishing Login URLs), follows our strategy, i.e. it contains URLs of both legitimate login pages and phishing ones. Table 1 shows the distribution of the available URLs into each subset.

To the best of our knowledge, none of the works in the state-of-the-art use legitimate login URLs specifically. By using legitimate login URLs, our work not only reflects the real-world scenario but also shapes an unbiased dataset in terms of URL length. Table 2 include examples of URLs of each class in PILU-90K, where differences are noticeable between the legitimate index URLs and the other two classes. Specifically, the length of the different parts of the URLs and the usage of keywords like *login*, *signin* or *secure*, are the most remarkable ones. Figure 3 provides an overview of the distribution of the URLs length in the proposed subsets, where PLU-60K displays a more similar distribution between classes than the PIU-60K subset.

Apart from the number of features, PILU-90K recreates a challenging scenario for URL phishing detection. On the one hand, a quarter of the legitimate login forms URLs do not have a path, i.e. login forms were located on the homepages, matching its URL structure with the homepage samples. On the other hand, one out of seven samples from the phishing class does not have a path, so they will also match with the legitimate homepage samples, increasing the classification challenge, even for skilled humans.

IV. METHODOLOGY

In this paper, we compare the performance of machine learning and deep learning methods for URL phishing classification. Regarding ML techniques, we used for feature extraction the handcrafted features proposed by Sahingoz *et al.* [21] and (ii) statistical features using Term Frequency-Inverse Document Frequency (TF-IDF) combined with character N-gram. Concerning the DL techniques, we adopted the CNN models of Zhang *et al.* [43] and Kim [44].

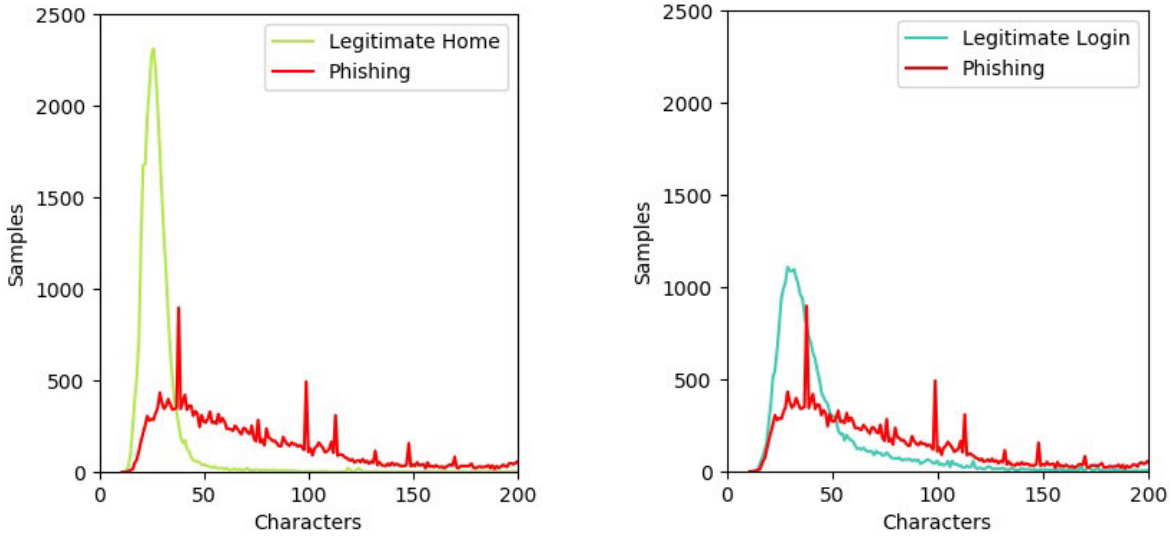
A. MACHINE LEARNING TECHNIQUES

Text classification based on supervised machine learning consists of three main stages: text preprocessing, text representation to convert the input text into a vector of features and a classifier. In this section, we explain the two techniques we used to extract features along with the evaluated classifiers.

[8]https://www.quantcast.com/products/measure-audience-insights/
[9]https://selenium.dev/projects/

TABLE 2. Examples of URLs within the different sets collected on PILU-90K dataset.

Class	URLs
Legitimate home	https://www.google.com/?gws_rd=ssl
	https://www.microsoft.com/es-es/
	https://www.netflix.com/es-en/
	https://www.wellsfargo.com/
	https://www.booking.com/
Legitimate login	https://accounts.google.com/signin/v2/identifier?hl=es&passive=true&continue=https%3A%2F%2Fw...
	https://login.live.com/login.srf?wa=wsignin1.0&rpsnv=13&ct=1572459869&rver=7.0.6730.0&wp=L...
	https://www.netflix.com/es/login
	https://www.wellsfargo.com
	https://account.booking.com/register?op_token=EgVvYXV0aCLABAoUdk8xS2Jsazd4WDl0VW4yY...
Phishing	http://google-ads-update-com.umbler.net/gmail/login/index.html
	https://login.microsoftonline.com/common/oauth2/authorize?client_id=4345a7b9-9a63-4910-a426-3...
	https://netflix-optusnetau.com/9fa04f87c9138de23e92582b4ce549ec/
	https://sycon.co.in/wellsf/https.wellsfargo.com.home/https.wellsfargo.com.home/wells-fargo-securit...
	https://azizi.groupbooking.co.in/



(a) PIU-60K subset (b) PLU-60K subset

FIGURE 3. Distribution of URL length across subsets. PIU-60K subset with a significant difference between classes. In contrast, PLU-60K subset has a closer length distribution over both classes.

For the handcrafted features, URLs were parsed using *tldextract*[10] library. Then, raw words are extracted from the different parts of the URL by splitting the string using a set of symbols (specifically, '/', '-', '.', '@', '?', '&', '=', '_'). After preprocessing, we extracted 38 features proposed by Sahingoz *et al.* [21] using URL rules and NLP features: the frequency of aforementioned symbols, number of digits in the domain, subdomain and path (see Figure 2) and their lengths. Other features are evaluated, such as the number of subdomains, domain randomness using the Markov Chain Model, whether it has a common TLD (Top Level Domain), whether 'www' or 'com' are on other places different from the TLD. From the raw words, the following metrics are extracted: maximum, minimum, average and standard deviation of the words length, number of words, compound words, words equals or similar to famous brands or a keyword like 'secure' or 'login', consecutive characters in the URL and the presence of Punycode.

Given these features, we trained and compared eight supervised classifiers commonly used in the related literature [28], [29], [45], [46]: Light Gradient Boosting Machine (LightGBM) [47], Extreme Gradient Boosting (XGBoost) [48], Adaptive Boosting (AdaBoost), Random Forest (RF), Support Vector Machines (SVM), k-Nearest Neighbours (kNN), Naïve Bayes (NB) and Logistic Regression (LR).

In NLP, another popular feature extraction technique is the TF-IDF algorithm [49], a statistical approach that gives more or less weight to a term depending on how many documents such term occur on, i.e. the higher the number of URLs a term occurs on, the lower the weight and vice-versa. A term in the TF-IDF algorithm can be either a word or N-gram of characters. Given that the URLs might not have word terms in common, we resorted to the character N-gram. Therefore, TF-IDF operates on the character N-gram level to find patterns of

[10]https://pypi.org/project/tldextract/

TABLE 3. Phishing datasets used in this work.

Dataset	Author	Year	Category	Legitimate samples	Phishing samples
PWD2016	Chiew et al. [18]	2016	A	12,550	15,000
1M-PD	Yuan et al. [19]	2017	A	500,000	500,000
Ebbu2017	Buber et al. [25]	2018	B	36,400	37,175
PIU-60K	This work	2020	A	30,000	30,000
PLU-60K	This work	2020	B	30,000	30,000

N consecutive characters of a given URL. Following the work of Al-Nabki *et al.* [50], we extracted grams between two to five characters, i.e. $N = [2, 5]$. The text preprocessing was limited to converting the text to a lower case. The extracted features were used to train an LR classifier given its good performance on similar noisy text tasks, such as File Name Classification [50], [51].

B. DEEP LEARNING TECHNIQUES

Besides the machine learning approaches, we explored the use of CNN to classify URLs [19], [41]. We selected the architectures of Zhang *et al.* [43] and Kim *et al.* [44], which operate at a character level.

The model of Kim *et al.* was originally built to function as a character-based language model. To use the model for URLs classification, we replaced the subsequent recurrent layers with a dense layer to perform a softmax operation over the classes. In contrast, the model of Zhang *et al.* did not require modifications to its architecture as it was intended for the text classification. It is worth mentioning that for both models, we did not carry out any text preprocessing step.

V. EXPERIMENTS AND RESULTS

A. DATASETS

To test the model robustness against URLs collected in different periods, we used the five phishing datasets shown in Table 3.

These datasets are grouped into two different categories depending on their recollection strategy: (i) category A: PWD2016, 1M-PD and PIU-60K collected legitimate samples by inspecting the top-visited domains and (ii) category B: Ebbu2017 and PLU-60K visited those websites and performed further actions: in the case of Ebbu2017, its authors retrieved the inner URLs and, in the case of PLU-60K, we looked for the login form page. Therefore most of the URLs include a path. Table 4 shows the distribution of sample structure within the datasets.

B. EXPERIMENTAL SETTINGS

Experiments are executed on an Intel Core i3 9100F at 3.6Ghz and 16 GB of DDR4 RAM. We used scikit-learn[11] and Python 3 for the implementation of the different experiments.

For the machine learning experiments, we empirically assign the parameters that returned the best accuracy on the three different phishing datasets. These parameters are shown in Table 5.

[11] https://scikit-learn.org/stable/

We used the averaged values of 10-fold cross-validation, reporting the accuracy (Eq. (3)), the F1-Score (Eq. (4)), the precision (Eq. (1)) and the recall (Eq. (2)) [21], [28], [34].

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{3}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

TP denotes the true positives, i.e., how many phishing websites were correctly classified. FP refers to the false positives and represents the number of legitimate samples wrongly classified as phishing. TN (i.e., the true negatives) denotes the number of legitimate samples correctly classified. Finally, FN represents the false negatives that represent the number of phishing websites misclassified as legitimate ones.

Regarding the clustering experiments, we used the same approach of Al-Nabki *et al.* [50] for text representation, as explained in Section IV-A and, for the clustering, we used the Agglomerative Hierarchical Clustering (AHC) [52]. The clustering process is repeated four times, and each time we initialized the AHC with the number n of the desired clusters, i.e. $n \in \{4, 5, 6, 7\}$.

VI. RESULTS AND DISCUSSION

A. MACHINE LEARNING AND DEEP LEARNING APPROACHES

In the following, we report the result of the designed machine learning classifiers using both handcrafted and automatic feature extraction techniques. Then, deep learning approaches are presented and compared with the previous ones. Finally, we proved the impact of using legitimate login URLs against the current state-of-the-art approach.

1) HANDCRAFTED FEATURE EXTRACTION

In this configuration, we extracted handcrafted features and benchmarked several classifiers, as explained in Section IV-A. Each model was trained and tested on each subset of the PILU-90K dataset. Table 6 reports the performance of each classier. It can be seen that XGBoost, LightGBM and RF outperform the rest of the classifiers on both subsets, obtaining 93.22%, 93.12% and 92.91% accuracy on PLU-60K, respectively. While for the PIU-60K sample subset, 94.63%, 94.67% and 94.42% accuracy were obtained, respectively. Results for the eight machine learning algorithms showed that Sahingoz *et al.* [21] descriptors achieve better performance on PIU-60K. Length-based features, the number of words and the presence of keywords enhance the performance when the difference between legitimate and phishing URLs is significant. Using the PLU-60K subset, such descriptors decrease their performance since their values are similar between classes.

TABLE 4. Phishing URLs datasets distribution.

		PWD2016	1M-PD	PIU-60K	Ebbu2017	PLU-60K
Legitimate URLs	w/o a path	**10,548 (84.00%)**	**471,728 (94.35%)**	**26,446 (88.15%)**	1,684 (4.62%)	6,693 22.31%
	w/ a path	2,002 (16.00%)	28,272 (5.65%)	3,554 (11.85%)	**34,716 (95.38%)**	**23,307 (77.69%)**
Phishing URLs		15,000	500,000	30,000	37,175	30,000

TABLE 5. Parameter configuration for the different models and datasets.

Algorithm	Parameter	PWD2016	Ebbu2017	PILU-90K
LightGBM	n_leaves	100	500	100
	objective	binary	binary	binary
XGBoost	n_estimators	100	100	100
AdaBoost	n_estimators	50	50	50
RF	n_estimators	250	350	250
kNN	k	1	1	3
SVM	kernel	rbf	rbf	rbf
	gamma	0.1	0.1	0.1
NB	algorithm	bernoulli	bernoulli	bernoulli
LR	solver	lbfgs	lbfgs	lbfgs

2) AUTOMATIC FEATURE EXTRACTION

In this experiment, we evaluate the classification pipeline that uses TF-IDF and character N-gram for feature extraction and LR for classification, as explained in Section IV-A. For each subset of the PILU-90K dataset, we trained a classification model and reported its performance. Automatic feature extraction methods have outperformed all the other methods in the F1-score, including those based on Deep Learning. For the PIU-60K, the classifier obtained an accuracy of 96.93%, while for the PLU-60K, accuracy was 96.50%. Hence, this model outperforms the benchmarked classifiers that depend on handcrafted features (see Table 6).

3) EVALUATION OF DEEP LEARNING-BASED PHISHING DETECTION MODELS

Similarly, we trained and evaluated the proposed CNN character-based models of both subsets of the PILU-90K dataset. We found that the model of Zhang *et al.* [43] has an accuracy of 95.22% on the PIU-60K subset and 94.10% on the PLU-60K one. The model of Kim [44] has a slightly better result with an average accuracy of 96.43% on the PIU-60K and 96.00% on the PLU-60K (see Table 6). Compared to machine learning algorithms, both CNN models obtained better results than handcrafted features but TF-IDF combined with N-gram [50] remains as the best classifier for the two proposed subsets.

4) IMPACT OF THE REPRESENTATION OF THE LEGITIMATE CLASS ON THE CLASSIFICATION

We assessed the impact on URL phishing classifiers when they are trained with samples where the legitimate class is represented with homepage URLs, e.g. PIU-60K. We trained 11 classifiers and reported their accuracy, as shown in Figure 4. Then, these models classified 30, 000 legitimate login URLs and their accuracy was reported again. It can be seen that all the models have suffered from a significant decrease in their accuracy. Al-Nabki *et al.* [50] model's

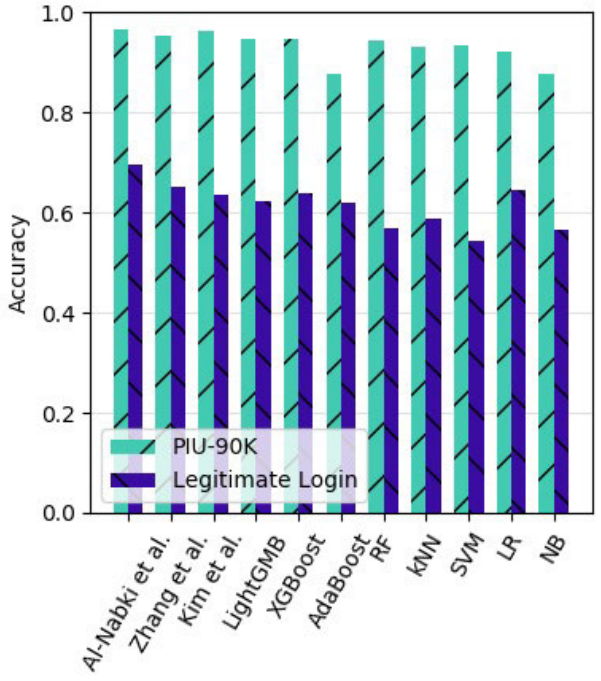


FIGURE 4. Accuracy of classification models trained on PIU-60K subset and the reported accuracy when classifying 30, 000 legitimate login URLs.

accuracy decreased 27% and was the most resilient with 69.50% accuracy. SVM decreased its accuracy up to 39.12% and obtained the worst result, 54.46% accuracy. CNN models of Zhang *et al.* [43] and Kim [44] obtained an accuracy of 65.13% and 63.50%, respectively. Furthermore, models based on handcrafted features, obtained the lowest accuracy, probably, due to the length-based features.

We observed that all models, including those trained with automatic features, misclassified more than 30% of the legitimate login URLs. These results can interfere with the application of the model in real-world applications since it presents a high false-positive rate. We argue that our TF-IDF and N-gram approach trained with PLU-60K can solve this issue since it can classify legitimate login samples with high accuracy as seen in Table 6. It should be noticed that this capability reduces overall accuracy in the advantage of reducing the false positives when users visit login pages.

B. ANALYSIS OF THE PERFORMANCE OF PHISHING MODELS OVER TIME

Recent machine learning proposals have reported good performance trained with PWD2016 and Ebbu2017 data sets. Since phishing attacks and, as a consequence, phishing

TABLE 6. Performance of the assessed algorithms on the subsets of PILU-90K datasets. The eight first rows correspond to handcrafted feature extraction methods, whereas the 9^{th} one corresponds to automatic feature extraction methods. The last two columns depict the results for the assessed deep learning models. All the results are given in %.

Algorithm	PIU-60K				PLU-60K			
	Precision	Recall	Accuracy	F1-Score	Precision	Recall	Accuracy	F1-Score
LightGBM	95.38	93.89	**94.67**	**94.63**	93.15	91.60	93.12	92.36
XGBoost	95.21	93.99	94.63	94.59	94.02	92.32	**93.22**	**93.16**
AdaBoost	94.18	91.72	93.03	92.93	89.24	85.82	87.74	87.50
RF	91.57	94.25	94.42	94.40	92.78	93.06	92.91	92.92
kNN	94.06	92.18	93.18	93.11	91.52	89.05	90.40	90.27
SVM	94.15	92.95	93.59	93.55	91.80	89.83	90.91	90.81
LR	93.57	90.91	92.33	92.22	86.64	81.87	84.62	84.19
NB	93.84	80.73	87.72	86.79	78.79	68.99	75.21	73.56
TF-IDF + N-gram	96.57	96.58	**96.93**	**96.93**	96.54	96.48	**96.50**	**96.51**
Zhang et al. [43]	95.93	94.57	95.22	95.24	92.12	95.90	94.10	93.97
Kim et al. [44]	95.22	97.57	96.43	96.38	95.96	96.02	96.00	95.99

websites' URLs get more and more sophisticated over time, we hypothesize that models trained with outdated datasets may decrease their performance when analyzing recent URLs.

To prove if this hypothesis is correct, we used PWD2016 and Ebbu2017 and the features from Sahingoz *et al.* [21] to train eight machine learning models (see Table 7) and test them using URLs from recent years. These datasets are 1M-PD from 2017, PIU-60K from 2020 and PLU-60K also from 2020. Among the proposed datasets we found two categories (see Section III). Datasets in category A were built using legitimate homepage URLs with no path, whereas in category B they include the path. For each category, we created a pipeline to avoid biased results. The first pipeline was focused on classifying URLs with no path, and we used category A datasets: PWD2016, 1M-PD and PIU-60K containing URLs collected in 2016, 2017 and 2020, respectively. In this pipeline, PWD2016 was used to train the eight machine learning algorithms and then it was evaluated using 1M-PD and PIU-60K. The second pipeline focused on classifying URLs with a path and, in this case, we used the datasets from category B: Ebbu2017 and PLU-60K, which contain URLs collected in 2017 and 2020, respectively. In this case, Ebbu2017 was used to train the proposed algorithms and then PLU-60K was utilized to test its performance.

From the experimental results shown in Table 7, all models struggled to endure over time and their performance decreased when tested on the following years' datasets. The model LightGBM obtained the best accuracy on both pipelines, but its results were the most affected over time, losing 10.42% and 30.69% accuracy on the first and second pipelines, respectively. On the other hand, SVM obtained the best results on recent datasets for the first pipeline, achieving 89.04% on the PIU-60K test, a 6.24% less than with the PWD2016 dataset used for training.

Overall results for the first pipeline, showed how a model trained with four years old datasets could not reach 90% accuracy, even when they obtained high performance on the base dataset. Moreover, the second pipeline, involving URLs classification with paths, also struggled to maintain performance on recent URLs.

TABLE 7. Phishing detection accuracy evolution over time (in %).

Training set	PWD2016			Ebbu2017	
Test set	PWD2016	1M-PD	PIU-60K	Ebbu2017	PLU-60K
LightGBM	97.60	91.42	87.18	95.94	65.25
XGBoost	97.47	91.65	87.59	95.27	65.75
AdaBoost	95.27	91.71	87.95	89.77	61.30
RF	97.32	91.72	88.15	95.69	64.02
kNN	95.49	90.02	86.42	92.55	58.92
SVM	95.28	91.87	89.04	93.05	63.43
NB	87.89	86.39	85.18	80.70	60.91
LR	93.37	89.07	86.95	87.90	58.40

C. CLUSTERING PHISHING URLs

In this experiment, we attempt to cluster the phishing URLs searching for patterns. By analyzing the obtained clusters, we did not identify significant relations among samples, despite the numbers of the clusters we tried. Nevertheless, when $n = 7$, we noticed associations between URLs but a further manual inspection of the clusters lead to uncertain conclusions. URLs were clustered due to similarities between different parts of the URL, i.e. similar domain or subdomain names were in the same cluster, but no further conclusions could be extracted.

Trying to look for phishing categories, we performed a term frequency analysis over the domain names of the URLs. First, we parsed the URL and obtained the domain using tldextract Python library.[12] Then, we sorted the results according to the domain frequency. We observed that the phishing class holds 12, 980 unique domain names, where 3, 543 of them were repeated using other subdomain or path. In order to identify the different categories, we performed a manual analysis of the 35 most common domains. We visited those domains and we evaluated the services provided on each domain, resulting in the six categories reflected in Table 8.

The first group is related to free subdomains, i.e. services that allow phishers to host their fake websites and make them accessible to the public. Typically, these services allow attackers to create a custom subdomain name to locate their website. Hence, this feature helps attackers in deceiving users by adding popular company names or using typosquatting and combosquatting techniques [53]. The main advantage of these hosting services is their price, as they have free plans

[12]https://pypi.org/project/tldextract/

TABLE 8. Most common phishing domains on PILU-90K dataset grouped into the spotted categories.

Type	Domain	Total domains
Free subdomain	000webhostapp	1415
	weebly	422
	umbler	398
	16mb	304
	godaddysites	197
	webcindario	134
	ddns	98
	joomla	76
	webnode	75
Cloud services	googleapis	125
	appspot	100
	sharepoint	97
	windows	90
	web	62
	xsph	59
	secureserver	55
	kl	49
Fake form	docs.google	713
	typeform	103
	forms.office	69
Standalone domains	update-information	71
	ticari	65
Social media	reddit	71
	steamcommunity	61
	twitter	61
Malware blog posts	imdb	550
	celestini	278
	fundraise	213
	ibm	195
	stackoverflow	110
	medium	107
	bandarrow	80
	toornament	65
	leetchi	55
	hatena	49

where phishers only introduce their email with no identity confirmation. Another advantage is the free SSL certificate they offer. However, the only disadvantage could be the limited free resource offered by these services, in terms of the bandwidth, storage and computation assets.

The second group comprehends cloud services. In this approach, phishers hire resources on different cloud platforms, such as Google or Azure, to host their phishing website with an SSL certificate. Some of these services provide fixed or random subdomains, and only the path can be edited. The main disadvantages of this strategy are the price and the fact that phishers have to provide payment information to hire the service.

Fake forms are common phishing methods. In these attacks, phishers use form platforms from Google, Microsoft or Typeform to look legitimate, using logos and messages to encourage the user to introduce their credentials. Companies have detected these issues and advise users not to introduce their personal information or credentials.

Social media and malware blog posts are reported on PhishTank to advise users from entering those sites. These domains usually offer free recent films for users to download and watch. These files are detected as malware by many commercial antivirus systems, such as Avast.

Finally, most of the dataset samples are related to standalone domains bought or compromised by phishers to host their websites. Within this category, some domains are used to host different campaigns of phishing over time. They get online on active campaigns and offline when such campaigns have finished or when they have been reported to blacklists.

VII. CONCLUSION

Phishing detection mechanism aims to improve current blacklist methods, protecting users from malicious login forms. Our work provides an updated dataset PILU-90K for researchers to train and test their approaches. This dataset includes legitimate login URLs which are the most representative scenario for real-world phishing detection.

We explored several URL-based detection models using deep learning and machine learning solutions trained with phishing and legitimate home URLs. The main advantage of our approach is the low false-positive rate when classifying this type of URL. Among the different evaluated models, TF-IDF combined with N-gram and LR algorithm obtained the best results with a 96.50% accuracy. In comparison with the current state-of-the-art, reviewed in Section II, our approach present three main advantages:

No dependence on external services. A limitation of the description methods that use features such as WHOIS domain age, page ranking on Google or Alexa or online blacklists, is their dependence on those services. Network slowdowns and service shortages can negatively impact analysis time, making real-time execution infeasible. Since phishing websites have a short lifespan [12], low detection times are required to warn users before accessing phishing websites.

Login website detection. Unlike other methods, which are trained with homepage URLs as representatives of the legitimate class, our model was trained with legitimate login websites. This ensures the correct classification of those websites. Therefore, our approach can be applied to the real-case scenario where users have to predict whether a login form page is legitimate or phishing.

Updated and real-world dataset. PLU-60K is focused on using updated legitimate login URLs. As demonstrated, models trained with old datasets were not able to endure their performance over time. We provide an updated phishing URL dataset for models to learn from nowadays phishing URLs and trends, which are crucial for real-world performance.

We demonstrated that phishing URL detection systems trained with legitimate land page URLs fail to classify legitimate login URLs correctly. The best-tested models could only classify 69.50% of these URLs correctly, which implies a high false-positive rate. For this reason, we recommend that a phishing detector, which intends to be used in a real situation, should be trained using *legitimate login* websites (such as PLU-60K) instead of homepages. The main drawback of using login websites for training is that, due to the similarity between phishing and legitimate samples, overall accuracy

is slightly reduced. The tradeoff against the state-of-the-art methods is still fair due to their high false-positive rate.

Different categories for current phishing attacks were identified by using a domain frequency analysis. While standalone and compromised domains were the most common approaches, free hosting services, cloud web servers and malware blog posts represent many current phishing attacks due to their cost and effectiveness for phishing campaigns.

Finally, we demonstrated that machine learning models using handcrafted URL features decreased their performance over time, up to 10.42% accuracy in the case of the LightGBM algorithm from the year 2016 to 2020. For this reason, machine learning methods should be trained with recent URLs to prevent substantial ageing from the date of its release. In the future, we will add more information about the samples into the analysis, such as the source code of the website and a screenshot of its content, which could be useful to increase the phishing detection performance. In addition, we will enlarge our dataset, including such information. Finally, observing that deep learning techniques and automatic feature extraction obtained promising results over traditional feature extraction, we intend to explore different URL codifications to improve detection performance.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of Nvidia Corporation for their kind donation of GPUs (GeForce GTX Titan X and K-40) that were used in this work.

DECLARATION OF COMPETING INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

[1] Statista. (2020). *Adoption Rate of Emerging Technologies in Organizations Worldwide as of 2020*. Accessed: Sep. 12, 2021. [Online]. Available: https://www.statista.com/statistics/661164/worldwide-cio-survey-operati%onal-priorities/

[2] R. De', N. Pandey, and A. Pal, "Impact of digital surge during COVID-19 pandemic: A viewpoint on research and practice," *Int. J. Inf. Manage.*, vol. 55, Dec. 2020, Art. no. 102171.

[3] P. Patel, D. M. Sarno, J. E. Lewis, M. Shoss, M. B. Neider, and C. J. Bohil, "Perceptual representation of spam and phishing emails," *Appl. Cognit. Psychol.*, vol. 33, no. 6, pp. 1296–1304, Nov. 2019.

[4] J. A. Chaudhry, S. A. Chaudhry, and R. G. Rittenhouse, "Phishing attacks and defenses," *Int. J. Secur. Appl.*, vol. 10, no. 1, pp. 247–256, 2016.

[5] M. Hijji and G. Alam, "A multivocal literature review on growing social engineering based cyber-attacks/threats during the COVID-19 pandemic: Challenges and prospective solutions," *IEEE Access*, vol. 9, pp. 7152–7169, 2021.

[6] A. Alzahrani, "Coronavirus social engineering attacks: Issues and recommendations," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 5, pp. 154–161, 2020.

[7] *Phishing Activity Trends Report 3Q*, Anti-Phishing Working Group, International, 2017. Accessed: Sep. 12, 2021.

[8] *Phishing Activity Trends Report 1Q*, Anti-Phishing Working Group, International, 2021. Accessed: Sep. 14, 2021.

[9] R. Chen, J. Gaia, and H. R. Rao, "An examination of the effect of recent phishing encounters on phishing susceptibility," *Decis. Support Syst.*, vol. 133, Jun. 2020, Art. no. 113287.

[10] *Phishing Activity Trends Report 4Q*, Anti-Phishing Working Group, International, 2020. Accessed: Sep. 12, 2021.

[11] S. Bell and P. Komisarczuk, "An analysis of phishing blacklists: Google safe browsing, OpenPhish, and PhishTank," in *Proc. Australas. Comput. Sci. Week Multiconf.*, Feb. 2020, pp. 1–11.

[12] A. Oest, Y. Safaei, P. Zhang, B. Wardman, K. Tyers, Y. Shoshitaishvili, A. Doupé, and G.-J. Ahn, "Phishtime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists," in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 379–396.

[13] L. Li, E. Berki, M. Helenius, and S. Ovaska, "Towards a contingency approach with whitelist- and blacklist-based anti-phishing applications: What do usability tests indicate?" *Behaviour Inf. Technol.*, vol. 33, no. 11, pp. 1136–1147, Nov. 2014.

[14] N. Samarasinghe and M. Mannan, "On cloacking behaviours of malicious websites," *Comput. Secur.*, vol. 101, pp. 102–114, Feb. 2021.

[15] L. Halgas, I. Agrafiotis, and J. R. C. Nurse, "Catching the phish: Detecting phishing attacks using recurrent neural networks (RNNs)," in *Information Security Applications* (Lecture Notes in Computer Science), vol. 11897. Cham, Switzerland: Springer, 2020, pp. 219–233.

[16] R. S. Rao and A. R. Pais, "Jail-phish: An improved search engine based phishing detection system," *Comput. Secur.*, vol. 83, pp. 246–267, Jun. 2019.

[17] Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, "Systematization of knowledge (SoK): A systematic review of software-based web phishing detection," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2797–2819, 4th Quart., 2017.

[18] K. L. Chiew, E. H. Chang, C. Lin Tan, J. Abdullah, and K. S. C. Yong, "Building standard offline anti-phishing dataset for benchmarking," *Int. J. Eng. Technol.*, vol. 7, no. 4, pp. 7–14, 2018.

[19] H. Yuan, Z. Yang, X. Chen, Y. Li, and W. Liu, "URL2 Vec: URL modeling with character embeddings for fast and accurate phishing website detection," in *Proc. IEEE Int. Conf. Parallel Distrib. Process. With Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Dec. 2018, pp. 265–272.

[20] M. Sánchez-Paniagua, E. Fidalgo, V. González-Castro, and E. Alegre, "Impact of current phishing strategies in machine learning models for phishing detection," in *Computational Intelligence in Security for Information Systems Conference*, vol. 12676. Cham, Switzerland: Springer, 2021, pp. 87–96.

[21] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Syst. Appl.*, vol. 117, pp. 345–357, Mar. 2019.

[22] Y. Cao, W. Han, and Y. Le, "Anti-phishing based on automated individual white-list," in *Proc. 4th ACM Workshop Digit. Identity Manage. (DIM)*, 2008, pp. 51–59.

[23] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "PhishNet: Predictive blacklisting to detect phishing attacks," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–5.

[24] A. K. Jain and B. B. Gupta, "A novel approach to protect against phishing attacks at client side using auto-updated white-list," *EURASIP J. Inf. Secur.*, vol. 2016, no. 1, pp. 1–11, Dec. 2016.

[25] E. Buber, B. Diri, and O. K. Sahingoz, "NLP based phishing attack detection from URLs," in *Proc. Int. Conf. Intell. Syst. Design Appl.*, vol. 736, 2018, pp. 608–618.

[26] A. K. Jain and B. B. Gupta, "PHISH-SAFE: URL features-based phishing detection system using machine learning," in *Advances in Intelligent Systems and Computing*, vol. 729. Singapore: Springer, 2018, pp. 467–474.

[27] B. Banik and A. Sarma, "Lexical feature based feature selection and phishing URL classification using machine learning techniques," in *Proc. Int. Conf. Mach. Learn., Image Process., Netw. Secur. Data Sci.*, vol. 1241. Singapore: Springer, 2020, pp. 93–105.

[28] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," *Neural Comput. Appl.*, vol. 31, no. 8, pp. 3851–3873, Aug. 2019.

[29] Y. Li, Z. Yang, X. Chen, H. Yuan, and W. Liu, "A stacking model using URL and HTML features for phishing webpage detection," *Future Gener. Comput. Syst.*, vol. 94, pp. 27–39, May 2019.

[30] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: A content-based approach to detecting phishing web sites," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, 2007, pp. 639–648.

[31] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, "CANTINA+: A feature-rich machine learning framework for detecting phishing web sites," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 2, pp. 1–28, Sep. 2011.

[32] M. Moghimi and A. Y. Varjani, "New rule-based phishing detection method," *Expert Syst. Appl.*, vol. 53, pp. 231–242, Jul. 2016.

[33] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Sov. Phys.-Dokl.*, vol. 10, no. 8, pp. 707–710, 1966.
[34] M. A. Adebowale, K. T. Lwin, E. Sánchez, and M. A. Hossain, "Intelligent web-phishing detection and protection scheme using integrated features of images, frames and text," *Expert Syst. Appl.*, vol. 115, pp. 300–313, Jan. 2019.
[35] M. Rami, M. Lee, and T. Fadi, "UCI machine learning repository," Univ. Huddersfield, Huddersfield, U.K., Tech. Rep., 2015.
[36] L. Yang, J. Zhang, X. Wang, Z. Li, Z. Li, and Y. He, "An improved ELM-based and data preprocessing integrated approach for phishing detection considering comprehensive features," *Expert Syst. Appl.*, vol. 165, Mar. 2021, Art. no. 113863.
[37] F. Sadique, R. Kaul, S. Badsha, and S. Sengupta, "An automated framework for real-time phishing URL detection," in *Proc. 10th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2020, pp. 335–341.
[38] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, no. 4, Dec. 2013, pp. 3111–3119.
[39] M. Somesha, A. R. Pais, R. S. Rao, and V. S. Rathour, "Efficient deep learning techniques for the detection of phishing websites," *Sādhanā*, vol. 45, no. 1, pp. 1–18, Dec. 2020.
[40] A. Aljofey, Q. Jiang, Q. Qu, M. Huang, and J.-P. Niyigena, "An effective phishing detection model based on character level convolutional neural network from URL," *Electronics*, vol. 9, no. 9, pp. 1–24, 2020.
[41] A. Al-Alyan and S. Al-Ahmadi, "Robust url phishing detection based on deep learning," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 7, pp. 2752–2768, 2020.
[42] J. Zhao, N. Wang, Q. Ma, and Z. Cheng, "Classifying malicious URLs using gated recurrent neural networks," in *Proc. Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput.*, vol. 773, 2019, pp. 385–394.
[43] X. Zhang, J. Zhao, and Y. Lecun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2015, pp. 649–657.
[44] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1746–1751.
[45] K. L. Chiew, C. L. Tan, K. Wong, K. S. C. Yong, and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning-based phishing detection system," *Inf. Sci.*, vol. 484, pp. 153–166, May 2019.
[46] R. S. Rao, T. Vaishnavi, and A. R. Pais, "CatchPhish: Detection of phishing websites by inspecting URLs," *J. Ambient Intell. Humanized Comput.*, vol. 11, no. 2, pp. 813–825, Feb. 2020.
[47] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2017, pp. 3147–3155.
[48] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vols. 13–17, 2016, pp. 785–794.
[49] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Inf. Process. Manage.*, vol. 39, no. 1, pp. 45–65, 2003.
[50] M. W. Al-Nabki, E. Fidalgo, E. Alegre, and R. Aláiz-Rodríguez, "File name classification approach to identify child sexual abuse," in *Proc. 9th Int. Conf. Pattern Recognit. Appl. Methods*, 2020, pp. 228–234.
[51] C. Peersman, C. Schulze, A. Rashid, M. Brennan, and C. Fischer, "ICOP: Live forensics to reveal previously unknown criminal media on P2P networks," *Digit. Invest.*, vol. 18, pp. 50–64, Sep. 2016.
[52] W. H. E. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *J. Classification*, vol. 1, no. 1, pp. 7–24, 1984.
[53] J. Spaulding, S. Upadhyaya, and A. Mohaisen, "The landscape of domain name typosquatting: Techniques and countermeasures," in *Proc. 11th Int. Conf. Availability, Rel. Secur. (ARES)*, Aug. 2016, pp. 284–289.

MANUEL SÁNCHEZ-PANIAGUA received the B.Sc. degree in computer science and the M.Sc. degree in cybersecurity research from the University of León, in 2019 and 2021, respectively. He is currently a Researcher at the GVIS Group, University of León. His research interests include anti-phishing solutions, fraud e-commerce website detection, web scrapping, cyber threat intelligence, and machine learning.

EDUARDO FIDALGO FERNÁNDEZ received the M.Sc. degree in industrial engineering and the Ph.D. degree from the University of León, in 2008 and 2015, respectively. He is currently the Coordinator of the Group for Vision and Intelligent Systems (GVIS), whose objective is the research and development of solutions to problems related to cybersecurity for INCIBE (https://www.incibe.es/en), by using artificial intelligence. His current research interests include natural language processing, computer vision, and machine and deep learning.

ENRIQUE ALEGRE received the M.Sc. degree in electrical engineering from the University of Cantabria, in 1994, and the Ph.D. degree from the University of León, Spain, in 2000. He is currently the Head of the Research Group for Vision and Intelligent Systems (GVIS) and an Associate Professor with the Department of Electrical, Systems and Automation Engineering, University of León. His research interests include computer vision and pattern recognition applications to medical and industrial problems and more recently, machine learning and computer vision applications for crime control and prevention.

WESAM AL-NABKI received the M.Sc. degree in robotics and computer vision from the University of Burgundy, in 2013, and the Ph.D. degree from the University of León, in 2019. He is currently a Researcher at the Universidad de León. His research interests include natural language processing, machine learning, and deep learning and their application in computer forensics.

VÍCTOR GONZÁLEZ-CASTRO received the B.S. and Ph.D. degrees in computer science from the University of León, Spain, in 2006 and 2011, respectively. He is currently an Associate Professor with the Department of Electrical, Systems and Automation Engineering, University of León. His current research interests include computer vision, machine learning, and deep learning, applied to medical images, quality assessment, and cyber-security.