EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

Combining Lexical, Host, and Content-based features for Phishing Websites detection using Machine Learning Models

Samiya Hamadouche[1,*], Ouadjih Boudraa[1] and Mohamed Gasmi[1]

[1]LIMOSE Laboratory, Faculty of Science, University M'Hamed Bougara of Boumerdes, Algeria

Abstract

In cybersecurity field, identifying and dealing with threats from malicious websites (phishing, spam, and drive-by downloads, for example) is a major concern for the community. Consequently, the need for effective detection methods has become a necessity. Recent advances in Machine Learning (ML) have renewed interest in its application to a variety of cybersecurity challenges. When it comes to detecting phishing URLs, machine learning relies on specific attributes, such as lexical, host, and content based features. The main objective of our work is to propose, implement and evaluate a solution for identifying phishing URLs based on a combination of these feature sets. This paper focuses on using a new balanced dataset, extracting useful features from it, and selecting the optimal features using different feature selection techniques to build and conduct a comparative performance evaluation of four ML models (SVM, Decision Tree, Random Forest, and XGBoost). Results showed that the XGBoost model outperformed the others models, with an accuracy of 95.70% and a false negatives rate of 1.94%.

Received on 11 November 2023; accepted on 16 April 2024; published on 17 April 2024

Keywords: Phishing URLs detection, Machine learning algorithms, Classification, Lexical-based features, Host-based features, content-based features, Feature selection.

doi:10.4108/eetsis.4421

1. Introduction

Cybersecurity is all about protecting, preventing, and recovering all the resources that use the internet from cyber-attacks which aim to steal, damage, or intrude into the personal or organizational confidential data [1]. Phishing attacks are one of the most prevalent threats to the whole internet community from individual users to large corporations and even service providers [2]. Phishing attacks can be defined as a cyber threat in which attackers take advantage of users through imitating legal original websites. They aim at stealing sensitive data like passwords and bank statements.

The Anti-Phishing Working Group (APWG) contributes individual reports on phishing URLs and analyzes the regularly evolving nature and procedures of cybercrimes. According to the latest report [3], corresponding to the fourth quarter of 2022, APWG observed 1,350,037 total phishing attacks. This was up slightly from the third quarter, when APWG recorded 1,270,883 total phishing attacks. This was a new record and the worst quarter for phishing that APWG has ever observed. As illustrated in figure 1, looking across four years (2019-2022), APWG has seen steep increases, accelerating to more than 150% per year [3].

The principal reason beyond this notable rise is that malicious people exploited the widespread of COVID-19 pandemic. In fact, so as to restrain the periods of lockdown that foiled their ordinary daily tasks (work, shopping, studies, etc.), people all around the world had to turn to the Internet. As many services move online due to the pandemic, COVID-19-themed

*Corresponding author. Email: hamadouche.samiya@univ-boumerdes.dz

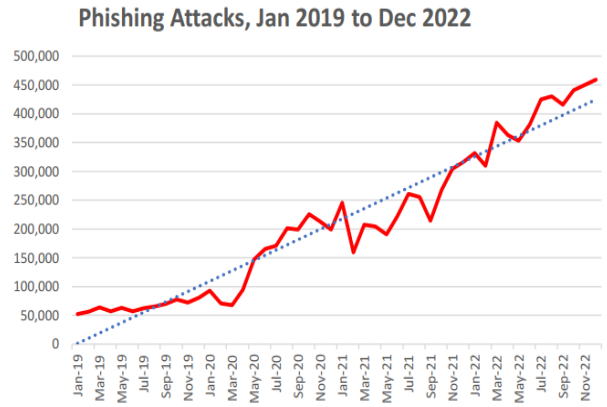


Figure 1. Phishing attacks growth from 2019 to 2022 (from [3])

cyber fraud is also growing. Therefore, the number of fake websites have been multiplied by cybercriminals so that they could deceive other victims. COVID-19 has not only changed people's overall behavior, but has also had a significant impact on psychological and mental health [4]. Thus, cybercriminals target victims' psychological vulnerabilities, taking advantage of COVID-19-related anxiety by manipulating emotional instabilities to enable cyber fraud [4]. Indeed, in phishing attacks, the attacker preys on human emotion and the urge to follow instructions in a flow [1].

Phishing is so omnipresent in the internet world that it has become a constant threat [1]. This has led researchers to pay more attention to this research domain. Therefore, various defense approaches to protect users against the phishing attacks were developed. Blacklists, which are mainly a database of Uniform Resource Locators (URLs) that have been recognized to be malevolent, are traditionally used to detect phishing websites. This technique is extremely fast and very easy to implement. However, these lists are not complete and cannot identify new generated URLs [5]. To overcome these issues, machine learning put forward a practical solution by new techniques to combat cybercrime through artificial intelligence using adequate algorithms.

Indeed, as stated earlier, machine learning techniques are playing a crutial role in various cybersecurity applications. Mainly for early detection and prediction of different attacks and threats such as spam [6, 7], fraud [8–10], malware [11–13], phishing [14, 15], intrusion [16, 17] and software vulnerability exploitation [18, 19].
Machine learning approaches use a set of URLs as training data, and based on the statistical properties, learn a prediction function to classify a URL as malicious or benign. This gives them the ability to generalize to new URLs unlike blacklisting methods [5].

The problem of detecting phishing URLs using machine learning techniques is not new but it is still relevant. It has been the subject of several research works (see section 2.3). However, the difference between the existing works lies in the way the problem was handled. It can be the variation of the extracted/selected features or even the used learning models.
In our present work, we address the same problem but from a different perspective. The key question we target to answer is the following: "What is the impact of combining several categories of selected features on the performance of the different models (phishing URLs detection classifiers)?". The key contributions of our work are summarized as follows:

- We collected our data (phishing / legitimate URLs) from different sources to build our own balanced and cleaned dataset.
- A total of 39 features were extracted from the constructed dataset (by parsing the URL or crawling the website). All these features belong to three different categories: lexical-based, host-based, and content-based.
- Several feature selection techniques were applied to select the most relevant ones for our problem. Four different subsets were obtained and used to evaluate the chosen classifiers.
- We conducted a comparative performance evaluation of 4 models (SVM, Decision Trees, Random Forest, and XGBoost) for classifying URLs as legitimate or phishing.
- We studied the impact of combining the content-based features with the other ones (lexical-based and host-based) on the performance of the different classifiers. We were able to conclude that, indeed, this combination of features allows us to efficiently improve the obtained results (in terms of accuracy and false negatives rate).

The rest of the paper is structured as follows. Section 2 recalls the general context in order to position our work. The proposed approach is discussed in section 3. Section 4 provides details about the implementation and analysis of experimental results. Finally, a conclusion to conclude the paper in section 5.

2. Background

2.1. Phishing attack

Phishing is a social engineering technique that, through the use of various methodologies, aims to influence the

target of the attack to reveal personal information such as an email address, username, password, or financial information. This information is then used by the attacker to the detriment of the victim [2].

The authors in [2], review a broad range of phishing techniques. According to them, the overall technique of phishing can be broken down into three constituent components that are interlinked:

- *The medium*: It represents the means by which the attacker communicates the phishing attack to the victim. These are: Voice, Short messaging service(SMS), multi-media messaging (MMS) and Internet.
- *The vector*: It is the channel through which the phishing attack is conducted. It is often limited by the medium. For example: Email, instant messaging, smishing (short message phishing), vishing (voice phishing), social networks and websites.
- *The technical approach*: It is the method deployed during the attack in order to gain access to the victim's personal details. For example: spear phishing, whaling, cross-site scripting, drive-by-download, wiphishing, browsers' vulnerabilities and clickjacking.

In our work, we are interested in website phishing attacks (called *URL phishing attack*). This kind of attack is carried out by using a hidden link [20]. The link holds the attackers' website. When the victim clicks the link, he is redirected to the attackers' website where his information is stolen.

URL is the abbreviation of Uniform Resource Locator, which is the global address of documents and other resources on the World Wide Web. A URL has two main components : protocol identifier (indicates what protocol to use) and resource name (specifies the IP address or the domain name where the resource is located) [21].
URL-based phishing attacks are mainly performed by embedding sensitive words or characters in a link that [22]:

- Mimics similar but misspelling words.
- Contains special characters for redirecting.
- Uses shortened URLs.
- Uses sensitive keywords which seem reliable.
- Adds a malicious file in the link and so on.

As per Tang et al [23], a generic phishing attack has four stages (illustrated in figure 2):

1. The attacker designs a fake website (i.e. phishing site) very similar to the users' trusted website,

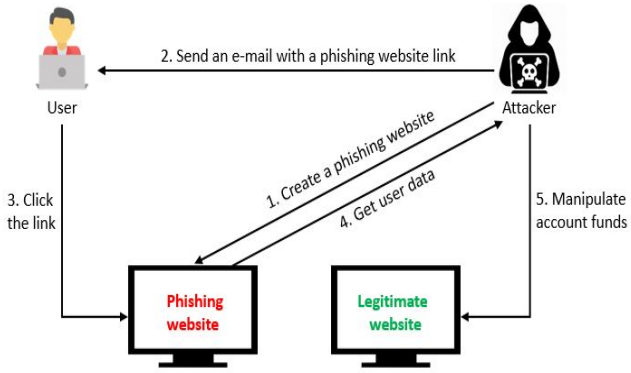


Figure 2. Phishing attack life cycle (adapted from [23])

2. The attacker tries to send the user a malicious link via email (the choice of the victim can be made intentionally or randomly).
3. The victim will click on the malicious link that has been received and redirected to the phishing site with a similar appearance to the real one that he knows. He will be persuaded to disclose his information as he does on the legitimate site while being on the fake one.
4. The users' information will be sent to the attacker who will exploit it to manipulate the user's accounts.

2.2. Machine learning based phishing URLs detection

The traditional approaches for phishing attack detection are not accurate and can recognize only about 20% of phishing attacks[1]. Thus, moving from traditional detection methods to more intelligent ones becomes crucial. Intelligent approaches, such as Machine Learning (ML) and Deep Learning (DL), have recently become widely popular in the research community to tackle challenges in various domains [24] such as cybersecurity, healthcare, medical imaging analysis, e-commerce, and social media.
ML approaches are popular for phishing websites detection and it comes down to a simple classification problem. They are mainly used for the prediction of new data output from historical data inputs (past experience).

Different classifiers may be used to detect a phishing attack. To train a machine learning model for detection, the input data must have features that are related to phishing and legitimate website classes. In machine learning based phishing detection, the characteristics (features) of the URLs are extracted and fed into the used algorithms (classifiers). I n g eneral, t he classifier

creates a model based on the information extracted from the training samples. Then, the suspected URL is evaluated according to this model to decide whether it is legitimate or not.

Several types of features can be used for malicious URLs detection (including phishing URLs). They are mainly categorized into [5]:

1. **Lexical-based features**: Obtained from the properties of the URL name (the URL string). It should be possible to identify the malicious nature of the URL based on its. The most commonly used lexical features include statistical properties of the URL string (length of URL, length of each component of the URL, number of special characters, etc.).

2. **Host-based features**: Obtained from the hostname properties of the URL. They allow us to know the location, identity, the management style and properties of malicious hosts. They may include: IP address properties, WHOIS information, domain name properties, connection speed, etc.

3. **Content-based features**: Obtained upon downloading the entire webpage. A lot of information needs to be extracted, and at the same time, safety concerns may arise. They include: HTML Document Level Features, JavaScript features, ActiveX Objects and feature relationships.

4. **Context features**: Represent the features of the background information where the URL has been shared (for example: social media platforms like twitter, facebook, etc.).

2.3. Related works

Several studies addressed the phishing URLs detection issue using machine learning algorithms. A number of literature reviews covering this subject exist [1, 21–23, 25, 26]. In table 1 we present a synthesis of some related works where we considered three classes of features: lexical, host and content based features. Few works have combined all three types of features. Most of the existing works have applied either one class based features only or a combination of at most two classes.

For the existing works summarized below (table 1), we will focus on the following points:

- *Dataset*: its size (number of phishing+legitimate URLs) and the nature of the data (balanced or unbalanced), which considerably influence the results. Indeed, a high accuracy with an imbalanced dataset is misleading.
- *Features*: their type (to which class they belong) and the total number of the selected ones
- *Models*: lists the used classifiers (KNN- K-nearest neighbors , SVM-Support Vector Machine , RF-Random Forest , DT-Decision Tree , GBoost-Gradient Boosting , NB-Naïve Bayes , *etc.*)
- *Performance*: accuracy and False Negatives Rate (FNR) which is the ratio of not detected phishing URLs.

However, we do not have the pretense to compare results considering the differences of the used: datasets, extracted features, applied algorithms, and the working environment. In our paper, we focused on investigating a combination of lexical-based, host-based and content-based features, which are essential in order to effectively detect phishing URLs.

3. Methodology

3.1. General process

In order to detect phishing URLs using machine learning, we followed a classical approach that is used in the majority of works dealing with the same problem. The process consists of the following four steps (Figure 3), which will be detailed in the next sections:

1. Data collection.
2. Data pre-processing : Data cleaning, feature extraction, data normalization, and feature selection.
3. Model training.
4. Evaluation of the solution.

3.2. Data collection

The first step consists of building the dataset on which the rest of the process will be based. Two types of URLs are needed: legitimate and phishing. Numerous datasets are available for web phishing detection. Most of the recent research has been conducted by using the real-world dataset, collected from online phishing databases such as PhishTank and Openphish [29]. Our work does not use pre-established datasets. In fact, we proceed to construct our own dataset from recognized, available and open source databases. Legitimate URLs were collected by crawling Alexa's top 1 million sites [37] which provides the most popular legitimate websites based on traffic data. While the phishing URLs were collected from Phishtank [38] which is the most used online community website. It provides millions of verified a nd n ewly discovered phishing URLs. This can help to stay ahead of emerging threats.

Table 1. Related works synthesis

Authors	Dataset size	Features	Models	Accuracy	FNR
Khan et al [27]	45343 (imbalanced)	Lexical (47)	Gboost, KNN, SVM, RF, Voting	96.60%	3.60%
Gupta et al [28]	19964 (balanced)	Lexical (9)	RF, KNN, SVM, Logistic regression	99.7%	0.30%
Mc Gahagan et al [29]	39183 (imbalanced)	Lexical + Content (133)	KNN, RF, AdaBoost, Gboost, XGboost, Bagging, Voting	98.03%	Not available
Li et al [30]	331622 (imbalenced)	Lexical + Host (49)	16 classifier	98.41%	Not available
Catak et al [31]	3231961 (imbalenced)	Lexical + Host (-)	GBoost, RF	98,60%	Not available
Korkmaz et al [32]	83857 (balanced)	Lexical + Host (48)	KNN, SVM, DT, RF, NB, AdaBoost, XGBoost	94,59%	5,31%
Mahjan et al [33]	36711 (balanced)	Lexical + Host (16)	DT, RF, SVM	97,14%	3,14%
Kumi et al [34]	1781 (balanced)	Lexical + Host + Content (11)	Classification Based on Association (CBA)	95.8%	1.35%
Li et al [35]	49947 (imbalanced)	Lexical + Host + Content (20)	GBDT, XGBoost, LightGBM	97.3%	4.46%
Aljabri et al [36]	66506 (balanced)	Lexical + Host + Content (15)	RF, NB, CNN, LSTM	96.01%	Not available
Our work	31980 (balanced)	Lexical + Host + Content (16)	SVM, XGboost, DT, RF	95.70%	1.94%

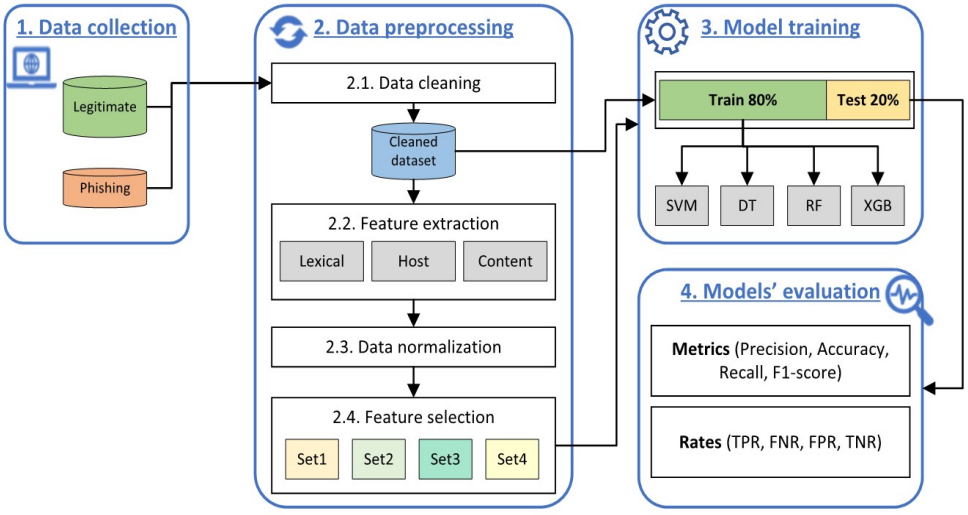


Figure 3. General process of the proposed solution

At the end of this collection step, we were able to obtain two datasets: one with 40000 legitimate URLs and another one including 38000 phishing URLs. The size imbalance of these resulting sets will be dealt with in the next step of the process (section 3.3).

3.3. Data preprocessing

The Preprocessing phase is fundamental in machine learning. It prepares the data for the application of classification t echniques. T his p hase i ncludes: data cleaning, feature extraction, data normalization, and feature selection. These steps are explained in more detail in the following sections.

Data cleaning. Constructing a cleaned representative dataset is more important than selecting a specific machine learning model, regardless of datasets size [39]. Thus, the cleaning phase is imperative so that we can ensure the good quality of our data to avoid false results. It consists in our case of the following operations:

1. **Standardization of the dataset format**: First, we had to organize the columns of the dataset which contained, initially, several useless columns. Indeed, we only need two of them. The first one, entitled "URL", contains URLs (the complete link) and the second one ,entitled "Label", can have two values: "1" when the URL is malicious (phishing) and "0" in case the URL is a legitimate one.

2. **Removing inaccessible URLs**: Due to the short life span of phishing URLs, it was necessary to delete all the URLs that are no longer accessible (not available in search engines). This was achieved by developing a python script which attempts to access each URL from the initial dataset. It checks wheather it is accessible by examining the response status code of a GET request and HTML content for XML sitemap links.

3. **Unsampling the dataset**: As previously indicated for the collected data, the ratios of the two classes "phishing" and "legitimate" are not equal (section 3.2). In order to prevent data bias (i.e. favor the majority class), we used the *randomized undersampling* technique. This technique is used to solve class imbalance problem in the dataset. It refers to the process of decreasing the number of samples in the majority class to balance it out with the minority class in the dataset [36]. In binary classification, using a balanced dataset is often needed, particularly when accuracy is utilized as the model evaluation metric [26]. Indeed, when learning models are trained on unbalanced datasets, their accuracy is misleading. In our case, the final obtained dataset is balanced between phishing URLs and legitimate ones (equal ratios).

Finally, we combined all the URLs obtained after cleaning in one dataset. The final dataset (cleaned data) gathers 31980 different URLs. To prevent data bias, as explained before, we have used legitimate and phishing URLs in an equal proportion (i.e. 15990 for each)

Feature extraction. This step is based on the cleaned list of URLs (obtained in the precedent step). Initially, according to the related works existing in the literature and the trendy set of features, addressing the same research problem, we have considered 39 features. We developed Python scripts for the automatic parsing and extraction of all features (the cleaned dataset contains only the "URL" and "label" columns, indeed, no predefined features). The results were saved into a .csv file.
The features' set adopted in our work can fall into three categories:

- Host-based features (5 features: F1-F5): derived from the HTTP headers of the responses to GET requests.

- Lexical-based features (11 features: F6-F16): collected through the lexical scanning of the URL string.

- Content-based features (23 features: F17-F39): mainly refer to HTML and JavaScript features. They were extracted by visiting the webpages and recording the content from the responses to GET requests.

The extracted features are resumed in table 2.

Data normalization. After analyzing our dataset, we noticed that all features have values between 0 and 1, except *URL length*, which has values way above 1 (on the order of tens and hundreds). This difference in data scale can have consequences on classifiers' performance. Moreover, the success of machine learning algorithms relies on the quality of the data to obtain a generalized predictive model of the classification problem [40]. Consequently, it is necessary to transform the values of features to a similar scale, which is called *normalization*. The purpose is to ensure that all features contribute equally to the model and avoid the domination of features with larger values. Among the existing normalization techniques, we applied *Min-Max normalization* to scale our data in the same range.
In this technique, the data is scaled to a range of [0,1] or [-1,1]. It converts the input value of the attribute to the range [*low*, *high*], by using the formula [41]:

$$x_{norm} = \frac{(high - low) * (x - minX)}{maxX - minX} \quad (1)$$

Where *low* and *high* are the minimum and maximum values of the attribute of the input dataset.

Table 2. List of the extracted features

Category	Features
Host-based features	F1: DNS Record F2: Website traffic F3: Age of Domain F4: End Period of Domain F5: Domain with copyright
Lexical-based features	F6: URL length F7-F12: Number of special characters (@, -, _, ?, &, .) F13: Is URL long F14: URL prefix suffix F15: Shortening the URL F16: URL depth
Content-based features	F17: IFrame Redirection F18: Status Bar Customization F19: Disabling Right Click F20: Abnormal Anchors F21: Page Empty F22: External CSS F23: Form Action F24: Popup window F25: Disable rightClick Bar F26: JavaScript Redirects F27: Use of Hidden Elements F28: Malformed HTML F29: Keylogging Indicators F30: Obfuscated JavaScript Libraries F31: Geolocation Tracking F32: Battery Status Tracking F33: Webcam and Microphone Access F34: Cookies Theft Detection F35: NoScript or AdBlocker Detection F36: Dynamic IP Detection F37: Phishing Kits or Frameworks F38: Email Harvesting F39: Use of Non-Standard Ports in URLs

Feature selection . The feature selection process consists of selecting a subset of the most relevant columns or features, in a dataset, for a given problem. Since irrelevant features can mislead machine learning algorithms and result in poor performance [36].
As all website features are not equally important to detect phishing websites, making use of relevant feature selection techniques is crucial to improve the machine learning model accuracy to speed up the time taken for training and testing as well as to address overfitting issues [26]. In our work, we considered several feature selection techniques to select with each of them a subset of the most relevant features. The obtained features' subsets will be all considered in the experimentation phase (section 4) in order to study the impact of feature selection on the classifiers' performance (section 4.3). The considered feature selection techniques are resumed as follows:

- **Correlation**: is a statistical term used to measure how strongly and in what direction two features are related. It is used to find the association between all the features and the target class feature. In our work, we considered two different correlation coefficients:
  - *Pearson Correlation Coefficient*. It is the most familiar measure for linear correlation. Its value ranges from -1 to 1, where -1 indicates a strong negative correlation, 0 indicates no correlation, and 1 indicates a strong positive correlation [42]. As per the standard literature, for a pair of variables (X,Y), the linear correlation coefficient 'r' is given in equation 2 below [43]:

$$r = \frac{\Sigma(X_i - \overline{X_i})(Y_i - \overline{Y_i})}{\sqrt{\Sigma(X_i - \overline{X_i})^2}\sqrt{\Sigma(Y_i - \overline{Y_i})^2}} \quad (2)$$

  - *Spearman correlation coefficient*. It determines a simple linear relationship between two variables and measures without dimensions. In other words, it is a coefficient that expresses the strength and direction of the relationship between two variables. Its value varies from -1 to +1. The relationship will be either negative or positive on one hand, and weak or strong on the other hand. The Spearman's coefficient of ranks correlation is given by the following formula [44]:

$$r_s = 1 - 6\sum d^2/n(n^2 - 1) \quad (3)$$

    where n is the number of ordered pairs and d represents the difference between the two ranks.

  The correlation coefficients (linear relationship between pairs) for all different features are displayed in a table called the *correlation matrix*. It is a powerful tool to summarize a large dataset and to identify and visualize patterns in the given data.

- **Feature importance**: It refers to techniques that assign scores to input features to a predictive model that indicates the relative importance of each feature when making a prediction. Feature importance scores can provide insight into the data as well as the model, and the basis for dimensionality reduction and feature selection.

This can improve the efficiency and effectiveness of a predictive model on the problem [45].

In our work, we considered different feature subsets in the experimentation phase. This is to compare and then determine which feature selection is the most representative for our problem by improving the performance of the classifiers. Table 3 resumes the chosen feature subsets.

3.4. Model training

Splitting the dataset into training and test sets. After building a clean dataset ready to be used for machine learning algorithms, we split it into two parts: one for training (train set) and one for testing (test set). When separating the data, usually, the majority of the data is used for training and the rest for testing in order to get a better performance of the learning models. Thus, we have separated our dataset of 31980 URLs according to the following ratio: training-test of 80-20 (i.e. 80% of the data for training and 20% for testing).

Choosing the learning algorithms. As previously stated, the problem of detecting phishing URLs comes back to a classic binary classification problem. We therefore used supervised machine learning techniques. There exists several algorithms for classification; in our case, we considered the following four algorithms (commonly used in the literature):

1. *Support vector machine (SVM)* [45] This classifier is a supervised learning helpful tool for regression and classification. It is a binary classification algorithm that separates the data points to find a hyperplane in case of many possible inputs. It uses decision functions, also known as support vectors, to perform classification. At least 4 types of kernels are used for classifcation: SVC with linear kernel, Linear SVC, SVC with RBF kernel, and SVC with the polynomial kernel.

2. *Decision tree (DT)* [23]. It is a popular machine learning algorithm, and the model logic is a tree structure. Each node in the decision tree is a feature; each stem presents a feature value and a possibility, and the last node presents the result. The more straightforward tree structure tends to have better performance.

3. *Random forest (RF)*[23]. It is an ensemble of decision trees for classification and regression. Random forests reduce the overfitting problem by classifying or averaging the output of individual trees in training processing. Therefore, random forests generally have higher accuracy than decision tree algorithms.

4. *eXtreme Gradient Boosting (XGBoost)* [32]. It is an algorithm based on gradient boosted decision trees that put speed and performance in the foreground. It aims to reduce errors in the previous tree by producing a new tree each time. The most important factor behind the success of XGBoost is its scalability in all scenarios. The system runs more than ten times faster than existing popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings [46].

3.5. Evaluation of the solution

The evaluation stage is a crucial step in choosing the best model. The selected models (algorithms) are trained on the training data to predict the results of the test data. Subsequently, the performance of each algorithm is evaluated using several metrics to compare their performance. The evaluation is based on the confusion matrix that summarizes the results of each model (correctly and incorrectly predicted instances). It has four outcomes namely true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

- **TP**: number of correctly predicted samples as phishing websites
- **TN**: number of correctly predicted samples as legitimate websites
- **FP**: number of incorrectly predicted samples as phishing websites
- **FN**: number of incorrectly predicted samples as legitimate websites

The metrics used to evaluate and compare the models are calculated from this matrix. They are summarized in table 4. In addition to these metrics for performance evaluation, we have to analyze the true positives rate (TPR), true negatives rate (TNR), false positives rate (FPR),and the false negatives rate (FNR). These rates and their calculation formulas are presented in table 5.

4. Experimental results

After the design of the solution (section 3), the implementation phase leads to the realization of our models. Then, come the phases of test and evaluation that will allow us to compare our models to choose the most performing one among them. Several Python libraries were used to implement the proposed solution, namely, Pandas, Numpy, Requests, BeautifulSoup, URLlib, Sklearn, and Matplotlib.

Table 3. The chosen feature subsets after feature selection

Subset	Selection technique	Size	List of selected features
Features_set1	Pearson correlation coefficient	16	F1.F4.F12-15.F17.F20-22. F24.F26.F30.F34.F35.F38
Features_set2	Spearman correlation coefficient	16	F1.F4.F12.F13.F15.F16.F17.F20-22.F24.F26. F30.F34. F35.F38
Features_set3	**Intersection** (Pearson Correlation coefficient, and the best common Most Important features)	7	F1.F13.F14.F17.F21.F22.F35
Features_set4	**Union** (Most Important features of all classifiers)	23	F1.F3.F4.F6.F8.F12.F14-17.F19-24.F26.F27.F29. F30.F34.F35.F38

Table 4. Evaluation metrics

Metric	Formula	Description
Precision	TP/(TP+FP)	Total number of URLs detected as phishing out of total phishing URLs
Accuracy	(TP+TN)/ (TP+TN+FN+FP)	Total number of overall correctly classified instances
Recall	TP/(TP+FN)	Total number of legitimate URLs classified as legitimate and phishing URLs classified as phishing.
F1-score	2*(precision*recall)/ (precision + recall)	The harmonic mean of precision and recall.

Table 5. Description of the used rates

Rate	Formula	Description
TPR	TP/(TP+FN)	The rate of phishing URLs that are classified as phishing
TNR	TN/(TN+FP)	The rate of legitimate URLs that are classified as legitimate
FPR	FP/(FP+TN)	The rate of legitimate URLs that are classified as phishing
FNR	FN/(FN+TP)	The rate of phishing URLs that are classified as legitimate

4.1. Experimental setup

As detailed in section 3.3, the initial dataset required a preprocessing phase in order to be ready for use in the training and testing phases. At the end of the preprocessing step, the dataset was splitted as follows: 25584 samples of URLs for the training set, and 6396 samples of URLs for the testing set. After that, the dataset was ready to be fed into the machine learning models, namely, the SVM, RF, DT and XGBoost. The experiments were carried out in a Google Research product called the *Google Colab* environment.

4.2. Hyperparameters tuning

Hyperparameters are model parameters whose values are set before training. *Hyperparameters tuning* is a process of choosing the right parameters for a classifier. It is an important step in the machine learning process. This will boost the accuracy of a classifier [28]. Therefore, a wrong choice of the hyperparameters' values may lead to wrong results and a model with poor performance.

In our work, we have performed hyperparameters tuning for all the classifiers by applying a *grid search* technique. It is an exhaustive exploration that tests all the combinations of hyperparameters given to the grid configuration (the domain of the hyperparameters) by calculating some performance metrics using cross-validation [47]. The point of the grid that maximizes the average value in cross-validation, is the optimal combination of values for the hyperparameters. Table 6 summarizes the hyperparameters forms applied for each model.

Table 6. Hyperparameters' settings applied for all classifiers

Model	Hyperparameter
SVM	Kernel=['rbf', 'sigmoid', 'linear']; C=[3, 5, 7, 8, 10]; random_state=12
DT	max_depth= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
RF	max_depth= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]; n_estimators= [100, 150, 200]

XGBoost	learning_rate= [0.01, 0.1, 0.2];
	max_depth= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
	n_estimators= [100, 150, 200];
	scale_pos_weight=[1, 2, 3]

4.3. Performance evaluation and results discussion

In this experimental part, we addressed three main questions:

1. How does the combination of the three features' categories (lexical, host and content) impacts the overall performance of the classifiers, compared to using content-based features only?
2. What are the most relevant features that improve the classifiers' performance?
3. What is the best model that gives the optimal values for metrics?

In this section, we will discuss the obtained experimental results in order to answer the above questions. All experiments were carried out on two categories of features: (1) Content-based only, (2) Combination of lexical+host+content based. Plus, we considered the four features' sets presented in the feature selection phase (section 3.3). For each configuration, we have used 6 396 instances as testing data each time.

We evauated our models on various evaluation metrics, namely, precision, accuracy, Recall, and F1-score. Tables 7, 8, 9, 10 show, separately for each feature' set, the confusion matrix and results obtained with the different classifiers for the two categories (content only or all).
We can notice that the results provided by all classifiers and for all feature' sets are clearly and considerably improved by the combination of lexical+host+content based features compared to the use of content-based features only. The answer to our first question is thus given. Therefore, from now on and throughout the rest of the discussion, we will use the term "results" to refer only to those results obtained by the lexical, host, and content-based feature combination.

In order to better visualize the results to be considered, figure 4 illustrates the values of the metrics obtained for the different models for each set of selected features. We can notice that:

- In *features_set1* (Pearson correlation coefficient): The XGBoost and SVM classifiers give the best results compared with DT and RF. Moreover, the values of the 4 metrics are relatively close (around 94%) for XGBoost and SVM.
- In *features_set2* (Spearman correlation coefficient): This set of features gives the worst results compared to the other sets for all classifiers. However, the precision values are higher than those of other metrics for DT, RF and XGBoost classifiers.
- In *features_set3* (Intersection (Pearson Correlation coefficient, and the best common Most Important features)): The DT, RF and XGBoost classifiers give modest results for all metrics. While SVM is the worst classifier with the lowest results compared to the other ones.
- In *features_set4* (Union (Most Important features of all classifiers)): The XGBoost and RF classifiers perform very close results which are better than those of the DT in terms of recall and F1-score (for precision and accuracy the values are close). The SVM classifier gives low values for all metrics except recall, which is on a par with the other classifiers.

In addition to the precedent evaluation metrics and in order to address phishing attacks effectively, two sorts of misclassifications are expected to be reduced by the phishing website detection model [26]:

- *False Positives rate (FPR)*: blocking online users from accessing legitimate websites due to incorrectly classifying legitimate websites as phishing.
- *False Negatives rate (FNR)*: allowing online users to visit malicious websites due to incorrectly classifying phishing websites as legitimate.

In this work, our priority is to reduce the false negatives rate (FNR). Indeed, the fact of wrongly classifying a phishing URL as legitimate constitutes the most dangerous and costly case in terms of security. However, the false positive rate (FPR) should not be neglected either, as it represents false alarms that can be harmful to the user.

The obtained rates with different classifiers are presented in table 11 and illustrated in figure 5. We can notice that:

- *Feature-set1*: The SVM and XGBoost models correctly classify both phishing and legitimate urls (high TPR/TNR and low FNR/FPR). On the

Table 7. Experimental results from different classifiers with features_set1

Features	Model	Confusion matrix				Metrics			
		TP	FP	FN	TN	Precision	Accuracy	Recall	F1-score
Content	SVM	2088	816	1110	2382	71,90	69,89	65,29	68,44
	DT	1690	249	1508	2949	87,16	72,53	52,85	65,80
	RF	1696	247	1502	2951	87,29	72,65	53,03	65,98
	XGB	1696	249	1502	2949	87,20	72,62	53,03	65,95
Lexical+ Host+ Content(16)	SVM	3039	217	159	2981	93,34	94,12	95,03	94,17
	DT	2774	750	424	2448	78,72	81,64	86,74	82,53
	RF	2774	750	424	2448	78,72	81,64	86,74	82,53
	XGB	3136	213	62	2985	93,64	95,70	98,06	95,80

Table 8. Experimental results from different classifiers with features_set2

Features	Model	Confusion matrix				Metrics			
		TP	FP	FN	TN	Precision	Accuracy	Recall	F1-score
Content	SVM	2087	823	1111	2375	71,72	69,76	65,26	68,34
	DT	1657	230	1541	2968	87,81	72,31	51,81	65,17
	RF	1645	213	1553	2985	88,54	72,39	51,44	65,07
	XGB	1682	278	1516	2920	85,82	71,95	52,60	65,22
Lexical+ Host+ Content(16)	SVM	2103	831	1095	2367	71,68	69,89	65,76	68,59
	DT	2224	253	974	2945	89,79	80,82	69,54	78,38
	RF	2226	252	972	2946	89,83	80,86	69,61	78,44
	XGB	2719	251	979	2947	91,55	82,16	73,53	81,55

Table 9. Experimental results from different classifiers with features_set3

Features	Model	Confusion matrix				Metrics			
		TP	FP	FN	TN	Precision	Accuracy	Recall	F1-score
Content	SVM	1860	610	1338	2588	75,30	69,54	58,16	65,63
	DT	1631	275	1567	2923	85,57	71,20	51,00	63,91
	RF	1623	292	1575	2906	84,75	70,81	50,75	63,49
	XGB	1623	292	1575	2906	84,75	70,81	50,75	63,49
Lexical+ Host+ Content(7)	SVM	2853	625	345	2573	82,03	84,83	89,21	85,47
	DT	2797	230	401	2968	92,40	90,13	87,46	89,86
	RF	2731	275	467	2923	90,85	88,40	85,40	88,04
	XGB	2745	284	453	2914	90,62	88,48	85,83	88,16

Table 10. Experimental results from different classifiers with features_set4

Features	Model	Confusion matrix				Metrics			
		TP	FP	FN	TN	Precision	Accuracy	Recall	F1-score
Content	SVM	2041	744	1157	2454	73,29	70,28	63,82	68,23
	DT	1829	267	1369	2931	87,26	74,42	57,19	69,10
	RF	2051	344	1147	2854	85,64	76,69	64,13	73,34
	XGB	2056	346	1142	2852	85,60	76,74	64,29	73,43
Lexical+ Host+ Content(23)	SVM	3079	744	119	2454	80,54	86,51	96,28	87,71
	DT	2834	201	364	2997	93,38	91,17	88,62	90,94
	RF	3174	200	124	2898	94,07	94,93	96,24	95,14
	XGB	3215	211	83	2887	93,84	95,40	97,48	95,63

other hand, the DT and RF classifiers give poor results, reflecting bad training of both classes.

- *Feature-set2*: The DT, RF, and XGBoost classifiers recognize legitimate urls (high TNR and low

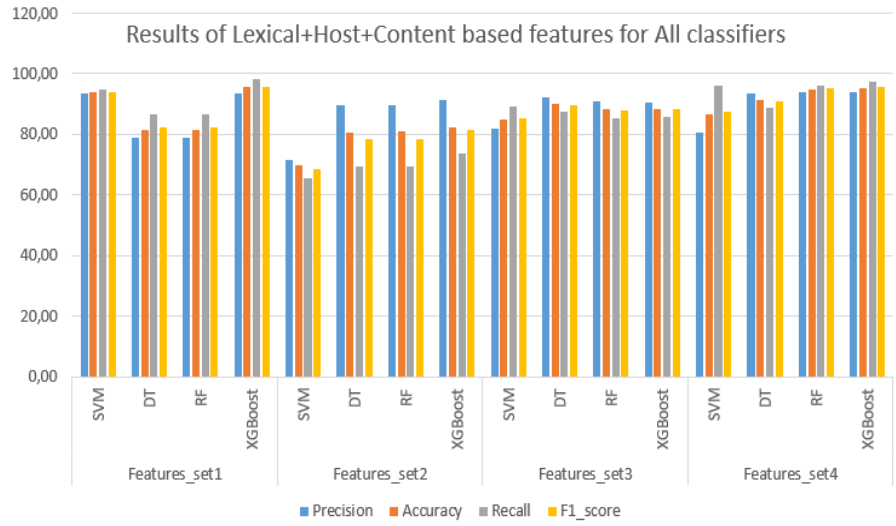


Figure 4. Experimental results from different classifiers with different features' sets (Lexical+host+content based features)

FPR) rather than phishing urls (low TPR and high FNR). This feature selection set therefore favors the legitimate class over the phishing class. Furthermore, the results of the SVM classifier indicate bad training.

- *Feature-set3*: The DT, RF, and XGBoost classifiers recognize legitimate urls (high TNR and low FPR) better than phishing urls (low TPR and high FNR). Like the feature-set2, this feature selection set favors the legitimate class over the phishing class. However, the SVM classifier recognizes the phishing class better than the legitimate class (FNR lower than FPR and TPR higher than TNR).
- *Feature-set4*: The RF and XGBoost models correctly classify both phishing and legitimate urls (high TPR/TNR and low FNR/FPR). On the other hand, the DT classifier favors the legitimate class over the phishing class (FPR lower than FNR and TNR higher than TPR). Therefore, the SVM classifier recognizes the phishing class better than the legitimate class (FNR lower than FPR and TPR higher than TNR).

Finally, to determine the best classifier for our phishing URL detection problem, we considered all metrics and rates. For us, an optimal model is the one that presents high metric values (precision, accuracy, recall and F1 score) but in a balanced way. In addition to that, the false positives and false negatives rates should be as low as possible. This can be interpreted as follows: the model makes accurate positive predictions, minimizes false positives and false negatives, and achieves a good balance between precision and recall. This indicates a good overall performance of the classification model.

Table 11. True/False Positives and Negatives Rates from different classifiers with different selected features

Features	Model	Rates			
		TPR	FNR	FPR	TNR
Set1	SVM	95,03	4,97	6,79	93,21
	DT	86,74	13,26	23,45	76,55
	RF	86,74	13,26	23,45	76,55
	XGBoost	98,06	1,94	6,66	93,34
Set2	SVM	65,76	34,24	25,98	74,02
	DT	69,54	30,46	7,91	92,09
	RF	69,61	30,39	7,88	92,12
	XGBoost	73,53	26,47	7,85	92,15
Set3	SVM	89,21	10,79	19,54	80,46
	DT	87,46	12,54	7,19	92,81
	RF	85,40	14,60	8,60	91,40
	XGBoost	85,83	14,17	8,88	91,12
Set4	SVM	96,28	3,72	23,26	76,74
	DT	88,62	11,38	6,29	93,71
	RF	96,24	3,76	6,46	93,54
	XGBoost	97,48	2,52	6,81	93,19

From tables 7 to 11, we have summarized the best results from classifiers that meet these criteria and therefore represent optimal models for solving our problem. Figure 6 illustrates this synthesis.

At the end of this experimental phase, we can conclude that the best model for our phishing URLs detection problem is the XGBoost classifier (precision=93,64%; accuracy=95,70%; recall=98,06%; F1-score=95,80%; FNR=1,94%; FPR=6,66%) using a balanced dataset and the feature-set1 (Pearson correlation)

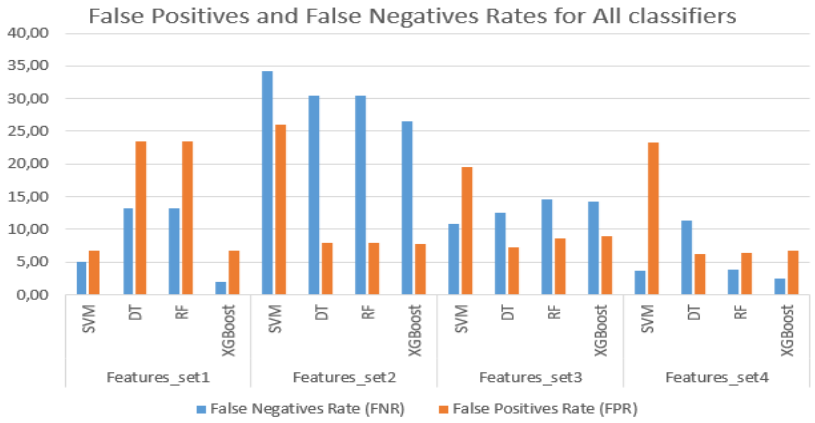
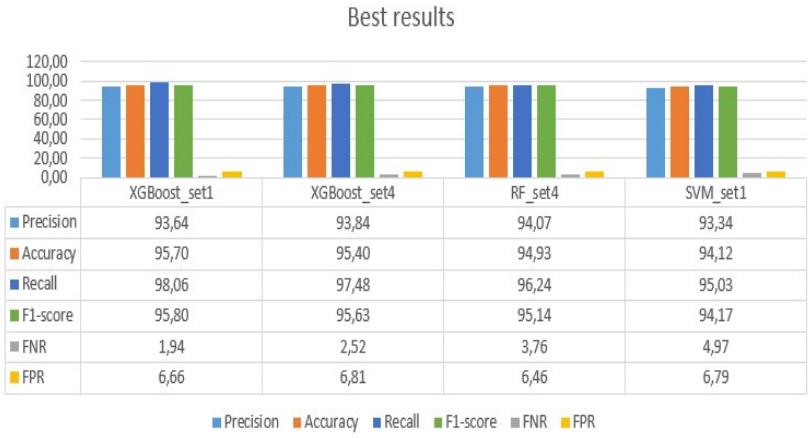


Figure 5. False positives and False negatives Rates for all classifiers (lexical+host+content based features)



	XGBoost_set1	XGBoost_set4	RF_set4	SVM_set1
Precision	93,64	93,84	94,07	93,34
Accuracy	95,70	95,40	94,93	94,12
Recall	98,06	97,48	96,24	95,03
F1-score	95,80	95,63	95,14	94,17
FNR	1,94	2,52	3,76	4,97
FPR	6,66	6,81	6,46	6,79

Figure 6. Top 4 classifiers given the best results

and combining three features' categories (lexical, host, and content).

5. Conclusion and future work

Phishing attacks have become one of the most prominent attacks faced by internet users, governments, and service-providing organizations [1]. Therefore, many research works dealt with this threat and tried to circumvent it because of its socio-economic impact. Machine learning techniques are proved to be efficient to overcome the problem of detecting phishing websites. In this paper, we first f ocused o n features engineering and selection by applying various feature selection techniques. After that, the performance of different models (SVM, DT, RF, and XGBoost) was evaluated using a number of metrics, mainly: precision, accuracy, recall, F1-score, and false positives/negatives rates. The experimental results show the efficiency of the combination of the lexical-based, host-based, and content-based features together. Moreover, the XGBoost algorithm outperformed the others models, with an accuracy of 95.70% and a false negatives rate of 1.94%. Future work aims to enhance the overall performance and robustness of the phishing URLs detection solution. Based on the findings of the present paper, we believe that improved results can be obtained by emphasizing the following research avenues:

- Explore more classification algorithms as well as deep learning models on larger datasets which can provide more representative samples of phishing URLs. This may lead to improve the detection accuracy and model robustness.
- Apply other feature selection techniques such as evolutionary algorithms which are inspired by biological processes. This can potentially allow to discover novel feature combinations that enhance model performance.
- Study the impact of ensemble learning classifiers on phishing URL detection. They combine multiple base classifiers to improve predictive

performance by leveraging the diversity of single models.

- Apply cross validation techniques to enhance the robustness of the evaluation process.

References

[1] Basit, A., Zafar, M., Liu, X., Javed, A.R., Jalil, Z. and Kifayat, K. (2021) A comprehensive survey of ai-enabled phishing attacks detection techniques. *Telecommunication Systems* **76**: 139–154.

[2] Alabdan, R. (2020) Phishing attacks survey: Types, vectors, and technical approaches. *Future internet* **12**(10): 168.

[3] (2021), APWG Phishing Trends Report: 4th quarter 2022. https://docs.apwg.org/reports/apwg_trends_report_q4_2022.pdf. Accessed: September 2023.

[4] Ma, K.W.F. and McKinnon, T. (2022) Covid-19 and cyber fraud: Emerging threats during the pandemic. *Journal of Financial Crime* **29**(2): 433–446.

[5] Sahoo, D., Liu, C. and Hoi, S.C. (2019) Malicious url detection using machine learning: A survey. *arXiv preprint arXiv:1701.07179* .

[6] Alurkar, A.A., Ranade, S.B., Joshi, S.V., Ranade, S.S., Shinde, G.R., Sonewar, P.A. and Mahalle, P.N. (2019) A comparative analysis and discussion of email spam classification methods using machine learning techniques. In *Applied Machine Learning for Smart Data Analysis* (CRC Press), 185–206.

[7] Dada, E.G., Bassi, J.S., Chiroma, H., Adetunmbi, A.O., Ajibuwa, O.E. *et al.* (2019) Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon* **5**(6).

[8] Prusti, D., Padmanabhuni, S.H. and Rath, S.K. (2020) Credit card fraud detection by implementing machine learning techniques. In *Safety, Security, and Reliability of Robotic Systems* (CRC Press), 205–216.

[9] Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M. and Anderla, A. (2019) Credit card fraud detection-machine learning methods. In *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)* (IEEE): 1–5.

[10] Khatri, S., Arora, A. and Agrawal, A.P. (2020) Supervised machine learning algorithms for credit card fraud detection: a comparison. In *2020 10th international conference on cloud computing, data science & engineering (confluence)* (IEEE): 680–683.

[11] Ma, Z., Ge, H., Liu, Y., Zhao, M. and Ma, J. (2019) A combination method for android malware detection based on control flow graphs and machine learning algorithms. *IEEE access* **7**: 21235–21245.

[12] Rathore, H., Sharma, S.C., Sahay, S.K. and Sewak, M. (2022) Are malware detection classifiers adversarially vulnerable to actor-critic based evasion attacks? *EAI Endorsed Transactions on Scalable Information Systems* **10**(1).

[13] Grace, M. and Sughasiny, M. (2022) Malware detection for android application using aquila optimizer and hybrid lstm-svm classifier. *EAI Endorsed Transactions on Scalable Information Systems* **10**(1).

[14] Patil, D., Pattewar, T., Pardeshi, S., Punjabi, V. and Wagh, R. (2022) Learning to detect phishing web pages using lexical and string complexity analysis. *EAI Endorsed Transactions on Scalable Information Systems* **10**(1).

[15] Sahingoz, O.K., Buber, E., Demir, O. and Diri, B. (2019) Machine learning based phishing detection from urls. *Expert Systems with Applications* **117**: 345–357.

[16] Hindy, H., Brosset, D., Bayne, E., Seeam, A.K., Tachtatzis, C., Atkinson, R. and Bellekens, X. (2020) A taxonomy of network threats and the effect of current datasets on intrusion detection systems. *IEEE Access* **8**: 104650–104675.

[17] Dey, S., Ye, Q. and Sampalli, S. (2019) A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks. *Information Fusion* **49**: 205–215.

[18] Yin, J., Tang, M., Cao, J., You, M., Wang, H. and Alazab, M. (2022) Knowledge-driven cybersecurity intelligence: software vulnerability coexploitation behavior discovery. *IEEE transactions on industrial informatics* **19**(4): 5593–5601.

[19] Yin, J., Tang, M., Cao, J., Wang, H., You, M. and Lin, Y. (2022) Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning. *World Wide Web* : 1–23.

[20] Zamir, A., Khan, H.U., Iqbal, T., Yousaf, N., Aslam, F., Anjum, A. and Hamdani, M. (2020) Phishing web site detection using diverse machine learning algorithms. *The Electronic Library* **38**(1): 65–80.

[21] Sahoo, D., Liu, C. and Hoi, S.C. (2017) Malicious url detection using machine learning: A survey. *arXiv preprint arXiv:1701.07179* .

[22] Aung, E.S., Zan, C.T. and Yamana, H. (2019) A survey of url-based phishing detection. In *DEIM Forum*: G2–3.

[23] Tang, L. and Mahmoud, Q.H. (2021) A survey of machine learning-based solutions for phishing website detection. *Machine Learning and Knowledge Extraction* **3**(3): 672–694.

[24] Aljabri, M., Aljameel, S.S., Mohammad, R.M.A., Almotiri, S.H., Mirza, S., Anis, F.M., Aboulnour, M. *et al.* (2021) Intelligent techniques for detecting network attacks: review and research directions. *Sensors* **21**(21): 7070.

[25] Jain, A.K. and Gupta, B. (2022) A survey of phishing attack techniques, defence mechanisms and open research challenges. *Enterprise Information Systems* **16**(4): 527–565.

[26] Adane, K. and Beyene, B. (2022) Machine learning and deep learning based phishing websites detection: The current gaps and next directions. *Review of Computer Engineering Research* **9**(1): 13–29.

[27] Khan, H.M.J., Niyaz, Q., Devabhaktuni, V.K., Guo, S. and Shaikh, U. (2019) Identifying generic features for malicious url detection system. In *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)* (IEEE): 0347–0352.

[28] Gupta, B.B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A. and Chang, X. (2021) A novel approach for phishing urls detection using lexical based

machine learning in a real-time environment. *Computer Communications* **175**: 47–57.

[29] McGahagan IV, J., Bhansali, D., Pinto-Coelho, C. and Cukier, M. (2021) Discovering features for detecting malicious websites: An empirical study. *Computers & Security* **109**: 102374.

[30] Li, T., Kou, G. and Peng, Y. (2020) Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods. *Information Systems* **91**: 101494.

[31] Catak, F.O., Sahinbas, K. and Dörtkardeş, V. (2021) Malicious url detection using machine learning. In *Artificial intelligence paradigms for smart cyber-physical systems* (IGI Global): 160–180.

[32] Korkmaz, M., Sahingoz, O.K. and Diri, B. (2020) Detection of phishing websites by using machine learning-based url analysis. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (IEEE): 1–7.

[33] Mahajan, R. and Siddavatam, I. (2018) Phishing website detection using machine learning algorithms. *International Journal of Computer Applications* **181**(23): 45–47.

[34] Kumi, S., Lim, C. and Lee, S.G. (2021) Malicious url detection based on associative classification. *Entropy* **23**(2): 182.

[35] Li, Y., Yang, Z., Chen, X., Yuan, H. and Liu, W. (2019) A stacking model using url and html features for phishing webpage detection. *Future Generation Computer Systems* **94**: 27–39.

[36] Aljabri, M., Alhaidari, F., Mohammad, R.M.A., Mirza, S., Alhamed, D.H., Altamimi, H.S., Chrouf, S.M. *et al.* (2022) An assessment of lexical, network, and content-based features for detecting malicious urls using machine learning and deep learning models. *Computational Intelligence and Neuroscience* **2022**.

[37] Alexa: Web information company's website. https://www.alexa.com/. Accessed: March 2022.

[38] Phishtank—join the fight against phishing. https://www.phishtank.com. Accessed: September 2023.

[39] Althnian, A., AlSaeed, D., Al-Baity, H., Samha, A., Dris, A.B., Alzakari, N., Abou Elwafa, A. *et al.* (2021) Impact of dataset size on classification performance: an empirical evaluation in the medical domain. *Applied Sciences* **11**(2): 796.

[40] Cabello-Solorzano, K., Ortigosa de Araujo, I., Peña, M., Correia, L. and J. Tallón-Ballesteros, A. (2023) The impact of data normalization on the accuracy of machine learning algorithms: A comparative analysis. In *International Conference on Soft Computing Models in Industrial and Environmental Applications* (Springer): 344–353.

[41] Bhanja, S. and Das, A. (2018) Impact of data normalization on deep neural network for time series forecasting. *arXiv preprint arXiv:1812.05519* .

[42] Ratner, B. (2009) The correlation coefficient: Its values range between+ 1/- 1, or do they? *Journal of targeting, measurement and analysis for marketing* **17**(2): 139–142.

[43] Blessie, E.C. and Karthikeyan, E. (2012) Sigmis: A feature selection algorithm using correlation based method. *Journal of Algorithms & Computational Technology* **6**(3): 385–394.

[44] Ali Abd Al-Hameed, K. (2022) Spearman's correlation coefficient in statistical analysis. *International Journal of Nonlinear Analysis and Applications* **13**(1): 3249–3255.

[45] Brownlee, J. (2016) *Master Machine Learning Algorithms: Discover How They Work and Implement Them From Scratch* (Machine Learning Mastery).

[46] Chen, T. and Guestrin, C. (2016) Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*: 785–794.

[47] Belete, D.M. and Huchaiah, M.D. (2022) Grid search in hyperparameter optimization of machine learning models for prediction of hiv/aids test results. *International Journal of Computers and Applications* **44**(9): 875–886.