

## A hybrid features based malevolent domain detection in cyberspace using machine learning

Saleem Raja Abdul Samad<sup>1</sup>, Pradeepa Ganesan<sup>1</sup>, Amna Salim Rashid Al-Kaabi<sup>1</sup>, Justin Rajasekaran<sup>1</sup>,  
Murugan Singaravelan<sup>2</sup>, Peerbasha Shebbeer Basha<sup>3</sup>

<sup>1</sup>Department of Information Technology, College of Computing and Information Sciences, University of Technology and Applied Sciences, Shinas, Sultanate of Oman

<sup>2</sup>Department of Computer Science and Engineering, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, India

<sup>3</sup>Department of Computer Science, Jamal Mohamed College, Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India

### Article Info

#### Article history:

Received Aug 29, 2024

Revised Jun 8, 2025

Accepted Jun 20, 2025

#### Keywords:

Machine learning

Malicious URL

Natural language processing

N-gram

Phishing

Word2Vec

### ABSTRACT

The rise of social media has changed modern communication, placing information at our fingertips. While these developments have made our lives easier, they have also increased cybercrime. Cyberspace has become a refuge for modern cybercriminals to conduct destructive actions. Most cyberattacks are carried out through malicious links shared on social media platforms, emails, or messaging services. These attacks can have serious consequences for individuals and organizations, including financial losses, sensitive data breaches, and damage to reputation. Early identification and blocking of such links are crucial to protecting internet users and securing cyberspace. Current research uses machine learning (ML) algorithms to detect malicious hyperlinks based on observed patterns in uniform resource locators (URLs) or web content. However, cyberattack tactics are constantly changing. To address this challenge, this paper introduces a robust method that performs a fine-grained analysis of URLs for classification. Lexical and n-gram features are examined separately, with URL n-grams represented using Word2Vec embeddings. The results from hybrid feature sets are combined using a logistic regression (LR) model to increase overall classification accuracy. This robust method allows the system to use both the structural components of the URL and the fine-grained patterns obtained by the n-grams.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



### Corresponding Author:

Saleem Raja Abdul Samad

Department of Information Technology, College of Computing and Information Sciences

University of Technology and Applied Sciences

Al Aqar, Shinas, Sultanate of Oman

Email: saleem.abdulsamad@utas.edu.om

## 1. INTRODUCTION

Most cyberattacks originate from malicious links disseminated via email, social media posts, and instant messaging applications. These links direct users to harmful websites specifically designed to compromise their security. By clicking on these links, victims can suffer a range of consequences, including the attacker collecting personal data from the victim, compromising accounts, downloading and installing malware (viruses, ransomware, spyware, or trojans), and wreaking havoc on the situation. Compromised accounts can harm the reputation of individuals and the company's brands and may lead to financial losses. Moreover, compromised systems may be used as "zombies" to launch further attacks on other networks. Detecting such malicious links in cyberspace is challenging for internet users [1]. To automate the detection

process, researchers consider different elements of the uniform resource locator (URL) or website, including URL lexical features, n-gram features, website content, reputation, and visual similarity [2]. Lexical features analyze the structural components of the URL, such as the domain name, path, number of subdomains, and number of dots, to identify suspicious patterns. N-gram analysis goes deeper by examining the character-level details of the URL, uncovering hidden patterns, subtle character changes, and obfuscation techniques. URL analysis is generally less resource-intensive and computationally efficient than other methods, allowing for the quick classification of malicious URLs with minimal risk to the system. Web content analysis, which inspects textual content, images, videos, and structural elements like hypertext markup language (HTML) tags and scripts, offers more accurate detection but poses risks, as malicious content can potentially infect the system analyzing it. Reputation checks rely on external services to assess a website's trustworthiness based on factors like domain server details and its ranking at both national and global levels. However, the short lifespan of malicious websites makes this process time-consuming. Finally, visual similarity analysis compares screenshots of suspicious websites with legitimate ones using image processing techniques.

However, this method is highly resource-intensive and demands significant computational power. Current security systems often rely on a blacklisting or signature-based approach. In this method, URLs are checked against a blacklist database to determine whether they should be allowed or denied. If the URL is present in the blacklist, access is blocked; otherwise, it is permitted. This is an easy and fast method for detecting malicious URLs. However, it has significant drawbacks. The blacklist database needs constant updates to identify newly created malicious URLs, making it ineffective against new threats. Additionally, attackers can bypass these systems by minor modifications to malicious URLs, making them undetectable unless the exact URL or signature is listed [3]. An alternative method is rules-based detection, which applies generalized rules to identify malicious URLs. While this helps broaden detection, it requires extensive domain expertise to create accurate rules. Additionally, modern attacks often require more complex rules, making them difficult to implement and maintain [4]. As a result, many researchers prefer machine learning (ML) approaches. They can detect previously unknown attacks that do not match existing signatures. By continuously training on large datasets, these models can improve their accuracy and dynamically adjust to changing patterns of malicious behavior [5], [6]. The proposed system combines URL structural elements (lexical features) with URL character-level elements (n-gram features) to enhance the detection of malicious URLs. Lexical features include the URL structure, domain length, subdomains, special characters, and query parameters, which help identify patterns associated with phishing or malware sites. To capture more granular characteristics, the system also utilizes n-gram features, which break down the URL into sequences of characters. These n-gram features are represented using Word2Vec embeddings, an ML technique that transforms character sequences into dense vectors. This approach captures the semantic relationships between different character sequences, enabling the system to detect subtle patterns in malicious URLs that might be missed by conventional methods [7]. The system achieves improved classification performance by integrating lexical and n-gram features compared to existing ML-based methods.

The contributions in this paper are: i) utilized a balanced dataset from a common repository to ensure reliable evaluation; ii) extracted more sensitive lexical features from the URL dataset to enhance detection accuracy; iii) employed n-gram features with Word2Vec embeddings to identify finer, hidden patterns within URLs; and iv) integrated lexical and n-gram features (hybrid feature), leading to superior classification performance compared to existing ML-based methods. The remainder of the paper is organized as follows: section 1 outlines the significance of the problem and presents an overview of the proposed method. Section 2 reviews existing research in the problem domain. Section 3 details the proposed method, while section 4 presents the experimental results. Finally, section 5 concludes the paper.

## 2. RELATED WORKS

This section presents an overview of recent research in the problem domain, focusing on URL feature analysis. Joshi *et al.* [8] developed a malicious URL detection method that significantly enhances malicious URL detection. To distinguish between dangerous and benign URLs, the authors gathered over 5 million URLs from various sources and identified 23 distinct lexical features. These lexical features were merged with 1,000 trigram-based features to generate 1,023-dimensional vectors. Among the classifiers, the random forest (RF) model scored the highest accuracy (92%), indicating that it is the most effective classification method. Raja *et al.* [9] used the University of New Brunswick (UNB) 2016 dataset for feature extraction, identifying 27 features (9 new, 18 standard). After correlation analysis, 20 features were selected for testing classifiers like RF, k-nearest neighbor (KNN), support vector classifier (SVC), logistic regression (LR), and naïve Bayes (NB). RF achieved 99% accuracy. Ha *et al.* [10] developed a system to detect fraudulent websites that use ML methods such as RF, decision tree (DT), AdaBoost, and KNN. Researchers used a dataset of 213,345 URLs divided into five categories (benign, defacement,

phishing, malware, and spam) and 20 extracted features. The RF algorithm achieved the highest accuracy at 95.68% in detecting dangerous websites.

Afzal *et al.* [11] proposed a system using artificial neural network (ANN) and bidirectional encoder representations from transformers (BERT) to extract contextual embeddings from URLs, protecting users from malicious websites by categorizing URLs into spam, phishing, malware, defacement, and benign. They used Kaggle and UNB datasets with 172,000 URLs for training and classification. The model utilized BERT's tokenization and a 12-layer transformer encoder, achieving a high accuracy of 98% and outperforming traditional embedding methods like Word2Vec, FastText, and GloVe. Their approach excelled in precision, recall, and F-measure, all at 98%. Kumi *et al.* [12] proposed a classification-based on association (CBA) technique for detecting risky URLs by assessing URL features and webpage content. CBA constructs an accurate classifier using association rules from a training dataset, achieving 95.8% accuracy with minimal false positive and negative rates. It demonstrated strong reliability when tested against 1200 tagged URLs. Al-Haija and Al-Fayoumi [13] proposed an ML-based approach for detecting malicious URLs using the ISCX-URL2016 dataset of 57,000 samples, including binary and multi-class labels and 79 features. Their method first differentiates between benign and harmful URLs and then splits malicious URLs into five categories: defacement, malware, phishing, spam, and benign. The researcher used four ensemble learning approaches, with bagging trees achieving the highest accuracy in binary (99.3%) and multi-class (97.92%) classifications.

Raja *et al.* [14] proposed a method for detecting bogus links by analyzing linguistic features of URLs. Three natural language processing (NLP)-based vectorizers are tested with six distinct ML methods. Results demonstrate that the proposed method with count vectorizer+RF algorithm delivers higher accuracy (92.49%). Lee *et al.* [15] demonstrated that particle swarm optimization (PSO) improves URL feature selection for malicious URL detection. The researchers used a support vector machine (SVM) and NB to obtain 99% accuracy. Raja *et al.* [16] developed a method for vectorizing URLs for feature generation using NLP techniques such as word frequency and inverse document frequency. To classify risky hyperlinks, researchers used a weighted soft voting classifier with two steps of weight adjustment to increase accuracy. The approach was evaluated on two datasets, D1 and D2, outperforming base classifiers. The accuracy obtained was 91.4% for D1 and 98.8% for D2, proving that it is better than other methods. Table 1 summarizes the existing methods. The related work highlights the importance of using a hybrid feature approach that combines both lexical features and n-gram features of URLs, enhanced by Word2Vec representation. This robust method has proven to be effective in classifying malicious URLs.

Table 1. Summary of related works

Author(s)	Dataset	Feature set	Remarks
Joshi <i>et al.</i> [8]	Openphish, Alexa whitelists, internal FireEye sources	23 lexical features of URLs+1,000 trigram-based features of URLs	Trigrams alone may not fully capture the detailed characteristics of the URL string. Moreover, contextual representation is also necessary.
Raja <i>et al.</i> [9]	UNB 2016 dataset	27 lexical features of URLs	The lexical features of URLs alone may not suffice for a robust system. N-gram features can be used to reveal hidden patterns within the URLs
Ha <i>et al.</i> [10]	OpenPhish Phishtank, Zone-H, WEBSHAM-UK2007.	20 lexical features of URLs	The lexical features of URLs alone may not suffice for a robust system. N-gram features can be used to reveal hidden patterns within the URLs
Afzal <i>et al.</i> [11]	Dataset from Kaggle repository	Word Embedding of URLs	Time-intensive and computationally expensive
Kumi <i>et al.</i> [12]	OpenPhish, VxVault, URLhaus	Lexical and host-based features of URLs, web content features	URLs' limited lexical features are insufficient for developing a robust system. Moreover, web content analysis may affect the processing system.
Al-Haija and Al-Fayoumi [13]	ISCX-URL2016	79 URL features	URLs' limited features are insufficient for developing a robust system. N-gram features can be used to reveal hidden patterns within the URLs
Raja <i>et al.</i> [14]	Dataset from Kaggle repository	8 lexical features of URLs and n-gram features of URLs	URLs' limited lexical features are insufficient for developing a robust system. Moreover, n-gram requires contextual representation.
Lee <i>et al.</i> [15]	Curated dataset	9 features include lexical and host-based features	URLs' limited lexical and host-based features are insufficient for developing a robust system.
Raja <i>et al.</i> [16]	ISCX-URL2016, UNB Phishtank	Vectorized URLs	The domain name alone is insufficient for effectively classifying a URL. Furthermore, a contextual representation of the URL is required rather than exclusively depending on vectorized representations.

### 3. METHOD

Recent research has increasingly focused on the lexical analysis of URLs due to their risk-free nature and faster detection capabilities. However, this approach has some limitations. For instance, encoded or hidden characters can evade lexical analysis, and legitimate URLs with unusual patterns, such as excessively long URLs with multiple parameters, might be incorrectly flagged as malicious. URLs that closely mimic popular legitimate sites might be misclassified as safe. To address these issues, the proposed system incorporates n-gram analysis, a method in NLP that examines continuous sequences of characters or words. This technique helps to identify unusual or suspicious patterns within URLs more effectively. Combining lexical analysis and n-gram analysis makes the system more robust and performs better than existing methods. Figure 1 shows the architecture of the proposed system.

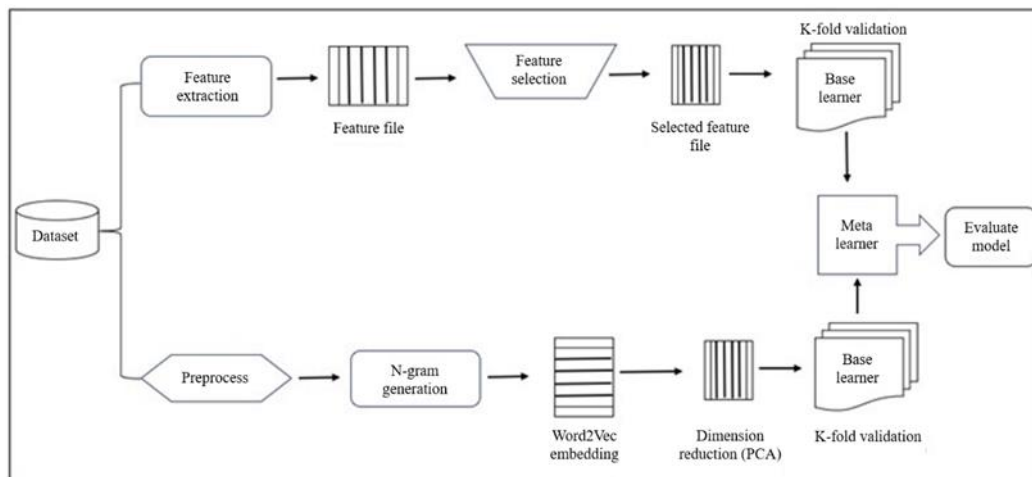


Figure 1. Architecture of the proposed method

#### 3.1. Dataset

The proposed method utilizes the full potential of the URL dataset. Tables 2 and 3 present a summary of the dataset, which includes benign and malicious URLs, including defacement, phishing, and malware URLs. The dataset used for the experiment was collected from Kaggle and PhishTank.

Table 2. Dataset details

Dataset	URLs
Kaggle [17]	Benign and malicious (phishing, malware, and defacement URLs)
Phishtank [18]	Malicious (phishing) URLs

Table 3. Dataset summary

Type	Count
Benign	15,530
Malicious	15,882

#### 3.2. Feature extraction

Feature extraction from URLs involves analyzing the structure of each URL to identify important characteristics that can reveal its underlying format and intent. This process helps in distinguishing between benign and malicious URLs based on specific patterns and tokens present within the URL string. Table 4 provides a detailed list of the lexical features extracted from the URL dataset, which were subsequently stored in a file for further analysis and model training.

##### 3.2.1. Subdomain

Cybercriminals often add multiple subdomains to create complex URL structures [19], [20] that mimic legitimate websites, deceiving users and security systems. For example, the <http://login.xbank.com.auth.phishmal.com> may appear legitimate at first glance, deceiving users into believing they are visiting a legitimate website. However, it's a malicious website.

##### 3.2.2. Punycode

Punycode is a way to represent Unicode characters in an American standard code for information interchange (ASCII)-compatible format, allowing non-ASCII characters to be used in domain names [21]. This encoding is essential for supporting internationalized domain names (IDNs), which allow users around the world to use native scripts in web addresses. For instance, the Punycode version of “appsdomain.com”

could be “xn--ppsdomain-9wb.com”, where a Cyrillic “a” replaces the standard Latin “a”, illustrating how deceptive lookalike domains can be created.

### 3.2.3. URL redirection

URL redirection is a technique that redirects users and search engines from one URL to another, directing traffic to a specified web page [22]. However, it can also be used for malicious purposes like phishing or malware distribution. Cybercriminals often use URL redirection to hide malicious links, directing users to bogus websites [23] that appear legitimate. Some keywords are most commonly used for URL redirection, such as redirect, target, and next.

### 3.2.4. Suspicious keywords

When analyzing potentially harmful URLs, identifying suspicious keywords is essential, as these terms often signal malware or phishing threats [24]. Such keywords typically include terms like “login,” “verify,” “update,” or brand names commonly exploited in social engineering attacks. In this experiment, a total of 63 keywords were identified and used as indicators of malicious intent to enhance detection accuracy.

### 3.2.5. URL entropy

In information theory, entropy measures uncertainty or randomness in a system. In the context of URLs, entropy can be used to measure their complexity and possible maliciousness by examining their length, character diversity, and overall structure [25]. High URL entropy indicates that the URL is more complex and potentially obfuscated.

Table 4. URL features

Feature	Description
url_len	Length of the URL
has_ssl	Check the SSL certificate for the URL
pres_subdom	Check the presence of a subdomain in the URL
age_dom	Age of the domain
url_entropy	Entropy of the URL
pres_punycod	Presence of Punycod in the URL
pres_short_url	Presence of the short URL
is_path_manipulation	Check the path manipulation
has_redirect	Checks the presence of the URL-redirection keywords in the URL
has_susp_keyword	Checks the presence of suspicious keywords in the URL
no_dom_segment	Number of sub-domains segments
avg_seg_len	Average length of the subdomain segments
path_entropy	Entropy of the URL path
path_len	Length of the path
path_splchr_count	Count the special characters in the path
dom_numeric	Is the domain numeric
sym_dots	Count dot symbols in the URL
sym_slash	Count slash symbols in the URL
sym_hyphen	Count hyphen symbols in the URL
sym_hash	Count hash symbols in the URL
sym_semicol	Count semicolon symbols in the URL
sym_and	Count and symbols in the URL
sym_underscr	Count underscore symbols in the URL
count_alpha	Count the alphabets in the URL
count_num	Count numbers in the URL
url_type	URL type (0-Benign, 1-Malicious)

### 3.3. Feature selection

The next step of feature extraction is feature selection. In the data preprocessing stage of ML, feature selection is an essential step that reduces dimensionality and chooses the most important features to improve model performance. This procedure helps remove irrelevant or redundant features, leading to overfitting and increased computing costs [26]. Using statistical testing, the SelectKBest technique is used for the experiment to select the features with the strongest correlation to the target variable. The analysis of variance (ANOVA) test determines whether there are significant differences in the means of different groups based on the features, allowing SelectKBest to rank features in order of significance.

### 3.4. Base learners

After feature selection, ML models were trained. Eight classification models were used for the experiment: SVC, LR, Gaussian Naïve Bayes (GNB), KNN, DT, RF, gradient boosting (GB), and extreme

gradient boosting (XGB) [16]. K-fold validation was used to measure the ML model's performance more accurately. Generally, a K-fold divides a dataset into K equally-sized folds or subsets. The value of K for our experiment is 10. One-fold is utilized as the validation set, and the remaining K-1 folds are used for training each iteration. This method ensures that every data point is used for training and validation, reducing bias and delivering a more accurate evaluation of the model's performance [27]. The model's predictions are passed to a meta-learner during validation to make the final decision.

### 3.5. Preprocess

In phase 2, the same dataset is used for n-gram generation. However, data preprocessing is required before generating n-grams [28]. This includes converting all URLs to lowercase and removing irrelevant characters, such as special symbols. Furthermore, frequent terms such as "https," "http," "ftp," and "www" are stripped away to provide cleaner and more informative n-gram analysis.

### 3.6. N-gram generation

To generate "n-grams" from URLs, text data is stripped into "n-grams," which are collections of adjacent characters of a predetermined length. To create n-grams for the experiment, sequences of 3 to 7 characters are extracted from URLs [14]. For example, given a URL like "mywebdom.com," 3-gram generation would produce sequences such as "myw," "ywe," and "web,". This process helps to capture both shorter and longer patterns within the URL.

### 3.7. Word2Vec embedding

In NLP, Word2Vec is the most common method for creating word embeddings, representing words as dense, continuous vectors in a high-dimensional space. Word2Vec contains two models, Skip-gram and continuous bag of words (CBOW) [29]. This experiment uses the CBOW model, which predicts a target word from its surrounding context. Representing URL n-grams using Word2Vec, which captures the semantic relationships between URL components to generate dense, meaningful vector representations. This technique allows for a more precise URL analysis with ML models.

### 3.8. Dimension reduction using principal component analysis

Dimensionality reduction is important in textual data preprocessing, especially after n-gram representation with Word2Vec. The n-gram format captures the context and co-occurrence of words by evaluating word sequences, resulting in high-dimensional feature spaces. This high dimensionality can present difficulties, such as higher computing costs and the possibility of overfitting. Principal component analysis (PCA) is a widely used dimensionality reduction approach that converts high-dimensional data into a lower-dimensional space while retaining as much variance as possible [30]. Using PCA after n-gram representation with Word2Vec embeddings can effectively reduce the dataset's complexity, allowing for more efficient processing and model performance.

- i) Matrix X represents the Word2Vec embeddings of a vocabulary, where each row corresponds to a word vector with  $d$  dimensions. If there are  $n$  words in the vocabulary, the matrix X will have dimensions  $n \times d$ , as shown in (1).

$$X = \begin{bmatrix} \vec{w_1} \\ \vec{w_2} \\ \vdots \\ \vec{w_n} \end{bmatrix} \quad (1)$$

Where  $\vec{w_i}$  is the Word2Vec embedding in the  $i^{\text{th}}$  word.

- ii) Subtract the mean of each dimension (feature) from the corresponding dimension of each word vector as shown in (2). This centers the data around the origin.

$$\bar{X} = X - \bar{X} \quad (2)$$

Where  $\bar{X}$  is the mean vector of the embeddings in X, computed as shown in (3).

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n \vec{w_i} \quad (3)$$

- iii) The covariance matrix C captures how different dimensions (features) of the Word2Vec embeddings co-vary. It is calculated as described in (4).

$$C = \frac{1}{n-1} \bar{X}^T \bar{X} \quad (4)$$

Where  $\bar{X}^T \bar{X}$  is the dot product of the centered matrix  $\bar{X}$  and its transpose, and  $\frac{1}{n-1}$  is a normalization factor.

- iv) Calculate the eigenvectors  $V$  and eigenvalues  $\lambda$  of the covariance matrix  $C$ . The eigenvectors represent the directions of the maximum variance, and the eigenvalues give the magnitude of the variance in those directions, as indicated in (5).

$$CV = \lambda V \quad (5)$$

- v) Sort the eigenvectors by their corresponding eigenvalues in descending order. Select the top  $k$  eigenvectors to form a projection matrix  $P$  as shown in (6).

$$P = [v_1 \ v_2 \dots v_k] \quad (6)$$

Where  $v_1, v_2, \dots, v_k$  are the top  $k$  eigenvectors.

- vi) Project the original Word2Vec embeddings  $\bar{X}$  onto the new subspace defined by the principal components as indicated in (7).

$$Z = \bar{X} P \quad (7)$$

Where  $Z$  is the new lower-dimensional representation of the Word2Vec embeddings, and  $Z$  will have dimensions  $n \times k$ , where  $k$  is the number of selected principal components.

### 3.9. Meta-learner

A meta-learner is an ML classifier that enhances predictive performance by integrating the results from multiple base learners. In this experiment, the meta-learner is an LR classifier that combines two base learners, one with a lexical URL representation and the other with an  $n$ -gram URL representation. The lexical URL representation captures the structure and components of a URL, whereas the  $n$ -gram representation analyzes sequential patterns within the URL. The meta-learner capitalizes on the capabilities of both base learners by integrating their predictions, allowing it to reach a more informed decision. This model can efficiently weigh each base learner's contribution using LR as the meta-learner, improving the classification accuracy [31]. In (8), represents the process of meta-learning.

$$y_{final} = \sigma (w_1 \cdot y_{lexical} + w_2 \cdot y_{n-gram} + b) \quad (8)$$

Where  $y_{final}$  is the final prediction for the meta-learner (LR);  $y_{lexical}$  is the prediction from the lexical URL representation base learner;  $y_{n-gram}$  is the prediction from the  $n$ -gram URL representation base learner;  $w_1$  and  $w_2$  are the weights the LR assigns to each base learner's output;  $b$  is the bias term in the LR; and  $\sigma$  is the sigmoid function, defined as in (9).

$$\sigma(z) = \frac{1}{1 + e^z} \quad (9)$$

The LR meta-learner combines the outputs  $y_{lexical}$  and  $y_{n-gram}$ , applies learned weights  $w_1$  and  $w_2$ , and passes the result through the sigmoid function to produce the final decision  $y_{final}$ , typically a probability or binary classification.

## 4. RESULTS AND DISCUSSION

The experiment was performed on a Windows i7 with Python and Jupyter Notebook. The popular scikit-learn package was utilized for ML algorithms. This experiment compared eight different ML algorithms. Each model was systematically trained and tested on a labeled URL dataset, allowing for a comprehensive comparison of their performance in accurately classifying malicious versus benign URLs. The results of the lexical representation of the URL are shown in Table 5. A total of 25 independent and 1 dependent feature was extracted from the URL, as detailed in Table 3. Using the SelectKBest method for feature selection, multiple ML algorithms were examined with a feature range of 15 to 25.

The results in Table 5 indicate that the number of features set at 20 and 25 yields nearly identical outcomes. In particular, with 20 features, the RF and XGB algorithms achieve accuracies of 99.22 and 99.36%, respectively. Lexical features alone could fail to recognize tiny yet vital patterns in malicious URLs,

such as specific character sequences or word combinations that appear frequently in such attacks. Word2Vec can map these sequences into continuous vector spaces, where semantically similar patterns are positioned closely together by representing the URL as n-grams. This improves the model's ability to detect anomalies and understand the relationships between URL fragments. Table 6 shows the experimental results for URL representation using n-gram in combination with Word2Vec. In this experiment, PCA was used to minimize the dimensionality of the vector space by selecting 35 to 45 components, maximizing the representation while keeping important information.

Table 5. Results of the lexical representation of URL

Number of features	ML algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
15	LR	86.60	92.13	80.60	85.98
	SVC	76.65	88.97	61.83	72.95
	GNB	84.14	90.20	77.27	83.23
	DT	98.50	98.46	98.59	98.53
	RF	99.15	98.95	99.40	99.17
	GB	98.22	97.55	98.99	98.26
	KNN	96.74	96.73	96.87	96.80
	XGB	99.20	99.10	99.33	99.22
20	LR	84.59	87.67	81.18	84.29
	SVC	76.63	88.91	61.84	72.94
	GNB	84.51	89.99	78.31	83.74
	DT	98.56	98.66	98.52	98.59
	RF	99.22	99.00	99.48	99.24
	GB	98.35	97.80	98.99	98.39
	KNN	97.08	96.24	98.09	97.16
	XGB	99.36	99.30	99.45	99.37
25	LR	84.49	88.07	80.50	84.10
	SVC	95.40	91.90	99.76	95.67
	GNB	83.39	89.68	76.15	82.36
	DT	98.58	98.63	98.58	98.60
	RF	99.23	99.02	99.48	99.25
	GB	98.36	97.81	99.00	98.40
	KNN	97.08	96.24	98.11	97.17
	XGB	99.37	99.30	99.46	99.38

Table 6 shows that the results obtained with 40 and 45 PCA components are almost identical. Therefore, PCA with 40 components was chosen for the experiment. Combining lexical and n-gram feature representations of URLs yields a more robust method for classifying URLs into malicious or benign. This hybrid method takes advantage of the URL's structural and sequential patterns, improving the model's capacity to detect threats more effectively. Table 7 presents the experimental results of the hybrid method (lexical and n-gram representation of URL). Figure 2 depicts the ROC-AUC Curve for the XGB classifier.

Table 6. Results of n-gram URL representation with Word2Vec and PCA (40 components)

Number of components	ML algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
35	LR	76.10	81.27	68.97	74.62
	SVC	77.21	88.07	63.94	74.07
	GNB	72.93	85.19	56.72	68.09
	DT	77.76	77.85	78.75	78.29
	RF	85.04	90.06	79.41	84.39
	GB	79.88	85.80	72.52	78.59
	AB	83.42	87.16	79.12	82.94
	XGB	84.07	86.60	81.31	83.87
40	LR	76.49	81.48	69.70	75.12
	SVC	77.26	88.24	63.88	74.10
	GNB	73.02	84.50	57.62	68.50
	DT	78.16	78.17	79.26	78.71
	RF	85.18	90.19	79.58	84.55
	GB	79.78	85.52	72.62	78.54
	KNN	83.53	87.37	79.12	83.03
	XGB	84.66	87.13	82.01	84.49
45	LR	76.43	81.51	69.52	75.03
	SVC	77.28	88.26	63.90	74.12
	GNB	73.00	83.91	58.17	68.68
	DT	77.87	77.99	78.82	78.39
	RF	85.13	90.22	79.43	84.48
	GB	79.87	85.58	72.75	78.64
	KNN	83.61	87.43	79.24	83.13
	XGB	84.78	87.27	82.12	84.61



Table 7. Experimental results of the hybrid method (lexical and n-gram URL representation)

ML algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
LR	84.68	84.85	84.74	84.67
SVC	78.12	79.97	78.35	77.87
GNB	84.54	85.04	84.65	84.51
DT	98.51	98.51	98.50	98.51
RF	99.30	99.30	99.30	99.30
GB	98.60	98.61	98.59	98.60
KNN	97.03	97.06	97.01	97.03
XGB	99.43	99.43	99.43	99.43

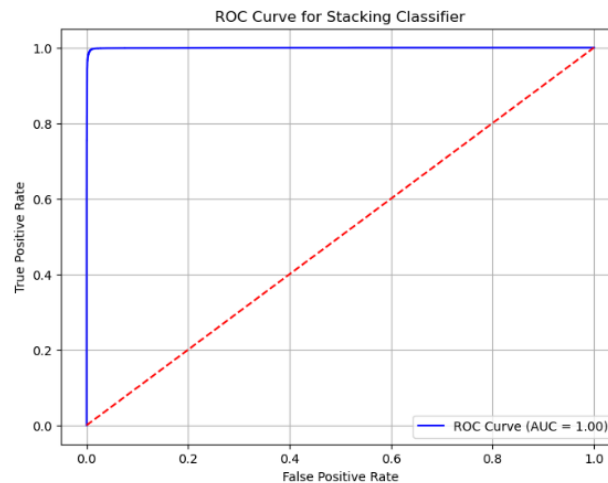


Figure 2. ROC-AUC curve for XGB classifier

The result shows that the RF and XGB give an accuracy of 99.30 and 99.43%, respectively. Table 8 and Figure 3 illustrate a performance comparison of the proposed hybrid method with existing methods. The results show that the proposed hybrid method surpasses other existing methods in accuracy.

Table 8. Performance comparison between existing methods and the proposed method

Author(s)	Accuracy (%)
Joshi <i>et al.</i> [8]	92
Raja <i>et al.</i> [9]	99
Ha <i>et al.</i> [10]	95.68
Afzal <i>et al.</i> [11]	98
Kumi <i>et al.</i> [12]	95.8
Al-Haija and Al-Fayoumi [13]	99.3
Raja <i>et al.</i> [14]	92.49
Lee <i>et al.</i> [15]	99
Raja <i>et al.</i> [16]	98.8
Proposed method	99.43

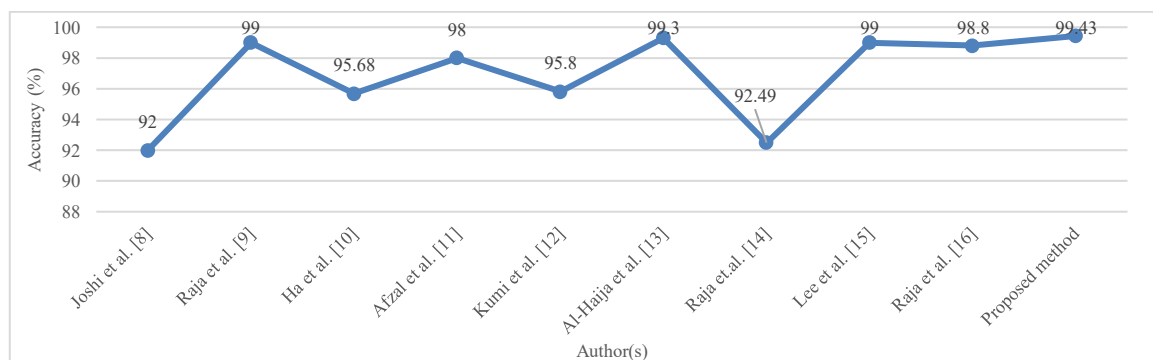


Figure 3. Graphical comparison of performance between existing methods and the proposed method

## 5. CONCLUSION

This study highlights the need for efficient detection systems to escalate cyberattack threats to various industries. The examination of related works has illuminated the limitations of existing methods, highlighting the necessity for our proposed approach. The proposed method leverages a balanced dataset to avoid bias. The proposed system enhanced detection accuracy by extracting sensitive lexical features and employing n-gram features with Word2Vec embeddings. Integrating these hybrid features resulted in superior classification performance, significantly outperforming existing methods. The experiment results reveal that the RF and XGB classifiers achieved impressive accuracy rates of 99.30 and 99.43%, respectively.

## FUNDING INFORMATION

This research project was funded by the University of Technology and Applied Sciences, Shinas, through the Internal Research Funding Program-2024, grant number (UTAS-Shinas-cy01-2024-002).

## AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Saleem Raja Abdul Samad	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Pradeepa Ganesan		✓				✓		✓	✓	✓	✓	✓		
Amna Salim Rashid Al-Kaabi	✓		✓	✓			✓			✓	✓		✓	✓
Justin Rajasekaran		✓			✓					✓				
Murugan Singaravelan					✓		✓			✓		✓		✓
Peerbasha Shebbeer Basha	✓	✓						✓	✓				✓	

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

## CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

## DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.





## REFERENCES

- [1] A. S. Raja, B. Sundarvadivazhagan, R. Vijayarangan, and S. Veeramani, "Malicious webpage classification based on web content features using machine learning and deep learning," *2022 International Conference on Green Energy, Computing and Sustainable Technology, GECOST 2022*, pp. 314–319, 2022, doi: 10.1109/GECOST55694.2022.10010386.
- [2] D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL detection using machine learning: a survey," *arXiv-Computer Science*, pp. 1-37, Aug 2019.
- [3] S. Liaquathali and V. Kadirvelu, "WCA: integration of natural language processing methods and machine learning model for effective analysis of web content to classify malicious webpages," *Journal of Advanced Research in Applied Sciences and Engineering Technology*, vol. 47, no. 1, pp. 105–122, Jun. 2024, doi: 10.37934/araset.47.1.105122.
- [4] S. Liaquathali and V. Kadirvelu, "Integration of natural language processing methods and machine learning model for malicious webpage detection based on web contents," *IAES International Journal of Robotics and Automation*, vol. 14, no. 1, pp. 47-57, Mar. 2025, doi: 10.11591/ijra.v14i1.pp47-57.
- [5] S. Sheikhi and P. Kostakos, "Safeguarding cyberspace: enhancing malicious website detection with PSO-optimized XGBoost and firefly-based feature selection," *Computers and Security*, vol. 142, 2024, doi: 10.1016/j.cose.2024.103885.
- [6] S. Abad, H. Gholamy, and M. Aslani, "Classification of malicious URLs using machine learning," *Sensors*, vol. 23, no. 18, 2023, doi: 10.3390/s23187760.




- [7] Y. I. Alzoubi, A. E. Topcu, and A. E. Erkaya, "Machine learning-based text classification comparison: Turkish language context," *Applied Sciences*, vol. 13, no. 16, 2023, doi: 10.3390/app13169428.
- [8] A. Joshi, L. Lloyd, P. Westin, and S. Seethapathy, "Using lexical features for malicious URL detection - a machine learning approach," *arXiv-Computer Science*, pp. 1-6, Oct. 2019.
- [9] A. S. Raja, R. Vinodini, and A. Kavitha, "Lexical features based malicious URL detection using machine learning techniques," *Materials Today: Proceedings*, vol. 47, pp. 163–166, 2021, doi: 10.1016/j.matpr.2021.04.041.
- [10] M. Ha, Y. Shichkina, N. Nguyen, and T. S. Phan, "Classification of malicious websites using machine learning based on URL characteristics," *Computational Science and Its Applications – ICCSA 2023 Workshops*, pp. 317–327, Jun. 2023, doi: 10.1007/978-3-031-37129-5\_26.
- [11] S. Afzal, M. Asim, M. O. Beg, T. Baker, A. I. Awad, and N. Shamim, "Context-aware embeddings for robust multiclass fraudulent URL detection in online social platforms," *Computers and Electrical Engineering*, vol. 119, 2024, doi: 10.1016/j.compeleceng.2024.109494.
- [12] S. Kumi, C. Lim, and S. G. Lee, "Malicious URL detection based on associative classification," *Entropy*, vol. 23, no. 2, pp. 1–12, 2021, doi: 10.3390/e23020182.
- [13] Q. A. Al-Haija and M. Al-Fayoumi, "An intelligent identification and classification system for malicious uniform resource locators (URLs)," *Neural Computing and Applications*, vol. 35, no. 23, pp. 16995–17011, 2023, doi: 10.1007/s00521-023-08592-z.
- [14] A. S. Raja, S. Peerbashab, Y. M. Iqbal, B. Sundarvadivazhagan, and M. M. Surputheen, "Structural analysis of URL for malicious URL detection using machine learning," *Journal of Advanced Applied Scientific Research*, vol. 5, no. 4, pp. 28–41, 2023, doi: 10.46947/joaasr542023679.
- [15] O. V. Lee *et al.*, "A malicious URLs detection system using optimization and machine learning classifiers," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 17, no. 3, pp. 1210–1214, 2020, doi: 10.11591/ijeecs.v17.i3.pp1210-1214.
- [16] A. S. Raja, S. Balasubramanian, P. Ganesan, J. Rajasekaran, and R. Karthikeyan, "Weighted ensemble classifier for malicious link detection using natural language processing," *International Journal of Pervasive Computing and Communications*, vol. 21, no. 1, pp. 26–42, 2025, doi: 10.1108/IJPC-09-2022-0312.
- [17] M. Siddhartha, "Malicious URLs dataset," *Kaggle*, 2021. Accessed: Oct. 04, 2024. [Online]. Available: <https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset>
- [18] PhishTank, "Developer information," *PhishTank*, 2009. Accessed: Oct. 04, 2024. [Online]. Available: [https://phishtank.org/developer\\_info.php](https://phishtank.org/developer_info.php)
- [19] C. Opara, Y. Chen, and B. Wei, "Look before you leap: detecting phishing web pages by exploiting raw URL and HTML characteristics," *Expert Systems with Applications*, vol. 236, 2024, doi: 10.1016/j.eswa.2023.121183.
- [20] F. Rashid, B. Doyle, S. C. Han, and S. Seneviratne, "Phishing URL detection generalisation using unsupervised domain adaptation," *Computer Networks*, vol. 245, 2024, doi: 10.1016/j.comnet.2024.110398.
- [21] CJ007, "Punycode technique use to prevent homograph-phishing attacks," *Medium*, 2023. Accessed: Oct. 03, 2024. [Online]. Available: <https://medium.com/@cjohngalvan/punycode-technique-use-to-prevent-homograph-phishing-attacks-87937a8bff2b>
- [22] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhonova, and A. A. Ghorbani, "Detecting malicious URLs using lexical analysis," *Network and System Security*, pp. 467–482, 2016, doi: 10.1007/978-3-319-46298-1\_30.
- [23] A. S. Rafsanjani, N. B. Kamaruddin, M. Behjati, S. Aslam, A. Sarfaraz, and A. Amphawan, "Enhancing malicious URL detection: a novel framework leveraging priority coefficient and feature evaluation," *IEEE Access*, vol. 12, pp. 85001–85026, 2024, doi: 10.1109/ACCESS.2024.3412331.
- [24] H. Tupsamudre, A. K. Singh, and S. Lodha, "Everything is in the name - a URL based approach for phishing detection," *Cyber Security Cryptography and Machine Learning*, pp. 231–248, 2019, doi: 10.1007/978-3-030-20951-3\_21.
- [25] K. Tran and D. Sovilj, "Advancing malicious website identification: a machine learning approach using granular feature analysis," *arXiv-Computer Science*, pp. 1-16, Sep. 2024.
- [26] S. R. A. Samad *et al.*, "Analysis of the performance impact of fine-tuned machine learning model for phishing URL detection," *Electronics*, vol. 12, no. 7, 2023, doi: 10.3390/electronics12071642.
- [27] J. M. Gorriz, R. M. Clemente, F. Segovia, J. Ramirez, A. Ortiz, and J. Suckling, "Is K-fold cross validation the best model selection method for machine learning?," *arXiv-Statistics*, pp. 1-40, Nov. 2024.
- [28] M. A. Tamal, M. K. Islam, T. Bhuiyan, and A. Sattar, "Dataset of suspicious phishing URL detection," *Frontiers in Computer Science*, vol. 6, 2024, doi: 10.3389/fcomp.2024.1308634.
- [29] S. V. Oprea and A. Băra, "Detecting malicious uniform resource locators using an applied intelligence framework," *Computers, Materials and Continua*, vol. 79, no. 3, pp. 3827–3853, 2024, doi: 10.32604/cmc.2024.051598.
- [30] K. K. Vasan and B. Surendiran, "Dimensionality reduction using principal component analysis for network intrusion detection," *Perspectives in Science*, vol. 8, pp. 510–512, 2016, doi: 10.1016/j.pisc.2016.05.010.
- [31] H. M. V. Kumari and D. S. Suresh, "Performance analysis of base and meta classifiers and the prediction of cardiovascular disease using ensemble stacking," *IDCIoT 2023 - International Conference on Intelligent Data Communication Technologies and Internet of Things, Proceedings*, pp. 584–589, 2023, doi: 10.1109/IDCIoT56793.2023.10053464.

## BIOGRAPHIES OF AUTHORS






**Saleem Raja Abdul Samad**     is a faculty member in the Department of Information Technology, University of Technology and Applied Sciences, Shinas, Sultanate of Oman. He received his Ph.D. degree in 2017 in the broad area of data mining and M.Tech.–Information Technology degree from Bharathidasan University, Tamil Nadu, India. He has over 17 years of teaching and research experience. Delivered technical talk on natural language processing, data science, mobile programming, and IoT. He has been supervising applied research projects and graduate student projects in cybersecurity. He has completed CCNA, HCAI, and Microsoft Associate certifications. He has published over 55 research papers in international and national journals, conferences, and book chapters. His research interests include cybersecurity, machine learning, deep learning, natural language processing, image processing, and the IoT. He can be contacted at email: [asaleemrajasec@gmail.com](mailto:asaleemrajasec@gmail.com) or [saleem.abdulsamad@utas.edu.om](mailto:saleem.abdulsamad@utas.edu.om).






**Dr. Pradeepa Ganesan**    is a faculty member in the Department of Information Technology at the University of Technology and Applied Sciences, Shinas, Sultanate of Oman. She holds her MCA degree from Madurai Kamaraj University, Tamil Nadu, India, her M.Phil. degree from Bharathidasan University, Tamil Nadu, India, and her Ph.D. from Vels Institute of Science, Technology and Advanced Studies, Tamil Nadu, India. She has 20 years of teaching experience. Her research interests include cybersecurity, natural language processing, machine learning, and deep learning. She can be contacted at email: [pradeepa.ganesan@utas.edu.om](mailto:pradeepa.ganesan@utas.edu.om).






**Ms. Amna Salim Rashid Al-Kaabi**    currently serves as the Head of the Department of Information Technology at the University of Technology and Applied Sciences, Shinas, Sultanate of Oman. She holds a master's degree in Management of Information Technology, which she earned from the University of Nottingham in the United Kingdom. With extensive experience in the IT sector, she is dedicated to advancing technology education and fostering innovation within her field. She can be contacted at email: [amna.alkaabi@utas.edu.om](mailto:amna.alkaabi@utas.edu.om).






**Justin Rajasekaran**    is working as a faculty member at the University of Technology and Applied Sciences, Shinas, Oman. He has authored several research papers and chapters in reputed national and international journals and presented research papers in national and international conferences under various domains, viz. Cloud computing in smart farming and its application, students' performance analysis using data mining techniques, mobile application testing, and malicious link detection using natural language processing. He has participated in various seminars, workshops, and faculty development programs. He has also participated in designing curriculum for AI and data science. He can be contacted at email: [justin.rajasekaran@utas.edu.om](mailto:justin.rajasekaran@utas.edu.om).



**Murugan Singaravelan**    is a faculty member in the Department of Computer Science and Engineering at Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Avadi, Chennai, Tamil Nadu, India. He earned his Ph.D. in 2022 from Anna University, Tamil Nadu, specializing in mobile ad hoc networks. With over 24 years of industrial experience and 6 years in teaching and research, he has delivered technical talks on the CONTIKI COOJA simulator and networks at Anna University. He also supervises graduate student projects in mobile networks and cloud computing. Additionally, he has completed CCNA and MCSA certifications. His research interests span cyber security, ethical hacking, machine learning, and deep learning. He can be contacted at email: [msing2k@gmail.com](mailto:msing2k@gmail.com) or [drsingaravelanm@veltech.edu.in](mailto:drsingaravelanm@veltech.edu.in).



**Peerbasha Shebbeer Basha**    is an Assistant Professor of the Department of Computer Science at Jamal Mohamed College, Affiliated with Bharathidasan University, Tamil Nadu, India. He has 16 years of teaching experience, including one year of international experience as a Senior Lecturer handling IT Courses at Southern Cross University, Australia. He holds his Ph.D. degree and is qualified with NET and SET in computer science and applications. He holds MCA, M.Phil., MBA, and M.Tech. He has received the Indo Asian Marvin Minsky Distinguished Innovative Researcher Award in Computer Science and the International Best Teacher Award. He has published more than twelve research articles in reputed international computer science journals. His area of research expertise includes machine learning and deep learning. He is a reviewer and editorial board member for many international journals related to computer science, engineering, and information technology. He can be contacted at email: [bashapeer2003@gmail.com](mailto:bashapeer2003@gmail.com).