Indonesian Journal of Computing, Engineering, and Design

Journal homepage: http://ojs.sampoernauniversity.ac.id/index.php/IJOCED

Optimizing Malware Detection and Prevention on Proxy Servers Through Random Forest and Lexical Feature Analysis

Meitro Hartanto Andalas Saputra[1], Dwi Pebrianti[2], Luhur Bayuaji[3], Rusdah[1]*

[1] Faculty of Information Technology, Universitas Budi Luhur, Indonesia
[2] Department of Mechanical & Aerospace Engineering, Faculty of Engineering, International Islamic University Malaysia, Malaysia
[3] Faculty of Data Science & Information Technology, INTI International University, Malaysia

Corresponding email: dwipebrianti@iium.edu.my

ABSTRACT

Malware has become a significant concern due to the increase in malicious websites hosting spam, phishing, malware, and other threats. This research aims to predict malware URLs using lexical features for feature extraction and random forest for classification. The dataset, sourced from kaggle.com, includes benign, phishing, spam, malware, and defacement URLs. To address data imbalance, random oversampling was applied for balanced training. Recursive feature elimination was used to optimize lexical features, testing various sets of features (10, 15, 19, 23, 29, 35) for classification accuracy, achieving 98% accuracy using 23 features. Validation tests with actual university network data confirmed this model's effectiveness, classifying malicious URLs in 9 minutes using 11,566 samples. URL filtering involved log analyzer tools capturing internet traffic during working hours over one month. Results suggest that this approach can efficiently classify malicious URLs and could be implemented for real-time detection in proxy server logs, aiding IT departments in preventing malware spread via web traffic.

ARTICLE INFO

Article History:
Received 15 Nov 2024
Revised 11 Mar 2025
Accepted 25 Mar 2025
Available online 14 Apr 2025

Keywords:
Lexical Features,
Malware Detection,
Proxy Server Logs,
Random Forest,
URL Classification.

1. INTRODUCTION

With the increasing dependency on the internet, organizations and individuals face a growing threat from cyberattacks, especially those originating from malicious websites. Malicious software, commonly called malware, is one of the most dangerous attacks on software that can damage the system by adding, deleting, or changing data or system software (Janabi & Altamini, 2020). The complexity of malware is one of the most serious cybersecurity threats (Hemalatha et al., 2021). Viruses, worms, and trojans are some examples of malware, where the

developer (cybercriminal) creates chaos on the system to steal important data, gain access, or cause a computer crash. According to recent statistics, the prevalence of malware attacks continues to rise, significantly impacting cybersecurity efforts worldwide.

Prevention is the process of trying to stop intruders from gaining access to system resources (Kizza, n.d.). In its application, the Intrusion Prevention System (IPS) monitors packet traffic in real-time (Abbas et al., 2023) based on malicious activity that matches a certain profile to provide warnings and block traffic passing through the network.

One of the primary vectors for malware delivery is websites hosting harmful content like phishing pages, spam, and other cyber threats. These websites often appear valid, making detecting them challenging for users and security systems. Traditional detection methods rely heavily on signature-based systems, which, although practical, struggle to keep up with the rapidly evolving nature of cyber threats.

Machine learning describes the ability of systems to learn from problem-specific training data to optimize the process of building analytical models and completing related tasks. It enables the emergence of intelligent systems with human-like cognitive capacity that enter business and personal life to improve decision-making for productivity, engagement, and employee retention (Kokila & Reddy, 2024).

This study aims to explore the application of machine learning, specifically the Random Forest algorithm, in detecting malicious URLs based on lexical features. By analyzing the structural components of URLs and employing techniques like Recursive Feature Elimination (RFE), this research aims to optimize feature selection to improve the classification accuracy of malicious websites. The ultimate goal is to enhance detection capabilities, allowing IT departments to mitigate the risks posed by malicious websites more effectively.

2. RELATED WORKS

Malicious URL detection has been an area of significant research interest in recent years, as the increasing prevalence of cyberattacks through malicious websites continues to pose a threat. Various machine-learning techniques have been explored to enhance the accuracy of detecting these threats.

For instance, some studies have applied the k-Nearest Neighbors (kNN) algorithm, focusing on URL classification based on lexical features and achieving moderate accuracy by optimizing distance measures (Reddy et al., 2023), (Pakhare et al., 2021), (Karajgar et al., 2024). Similarly, Random Oversampling (ROS) has been used to address imbalanced datasets in malicious URL detection, improving classification performance in scenarios where benign URLs vastly outnumber malicious ones (Pag et al., 2019).

A more advanced method frequently used is the Random Forest classifier, a robust ensemble technique known for its high accuracy and scalability in detecting malicious URLs. As discussed by Fawagreh et al. (Fawagreh et al., 2014), Random Forest has consistently outperformed other classifiers like Multinomial Naive Bayes and Decision Trees due to its ability to handle both categorical and continuous input variables effectively. Additionally, logistic regression and Support Vector Machines (SVM) have been evaluated as alternatives, particularly in the context of URL length and entropy features, demonstrating competitive results

(Dhingra et al., 2023) (Ahammad et al., 2022).

Integrating Natural Language Processing (NLP) techniques with machine learning models has recently gained traction. Convolutional Neural Networks (CNNs) have also been proposed to identify URL patterns and structure anomalies, leading to improved results in malware and phishing URL detection (Jasim & Farhan, 2023), (Alsaedi et al., 2023), (Pushpalatha & Vijaya, 2023).

Other notable approaches include cross-validation to ensure feature selection methods' robustness and Recursive Feature Elimination (RFE) to optimize feature sets for better classification results (Sharma & Yadav, 2021). Finally, Calderon et al. highlighted the importance of HTTPS characteristics in malicious website detection, utilizing machine learning to analyze encrypted web traffic data for anomaly detection (Calderon et al., 2018).

Additionally, Pektas and Acarman explored Hybrid Deep Learning, integrating CNN and LSTM models to improve detection accuracy by analyzing the URL structure and the sequence of characters (Ogbuagu et al., 2024). Wang et al. demonstrated the use of Graph Neural Networks (GNN) to capture complex relationships between URLs and their host information, offering significant improvements over traditional feature-based methods (Huang et al., 2023). Federated Learning has also been proposed by Lee and Hsieh, allowing collaborative training of malicious URL detection models across multiple organizations without sharing sensitive data, further enhancing detection accuracy while preserving privacy (Khramtsova et al., 2020).

These various methodologies show the evolving nature of machine learning-based malicious URL detection, with ensemble methods like Random Forest leading due to their adaptability and high accuracy. This study builds upon these approaches by optimizing lexical features using RFE, achieving improved classification accuracy through careful feature selection and comparison across different classifiers.

This study builds on these findings by optimizing feature selection using Recursive Feature Elimination (RFE) and applying Random Forest for classification. Unlike previous approaches, our method reduces computational complexity while maintaining high accuracy and combining lexical feature selection and Random Forest, which results in an efficient and scalable malware detection model suitable for proxy server environments.

3. MATERIALS AND METHODS

Figure 1 shows the process that was used in this study. A detailed explanation will be provided below in a sequence of subsections.

3.1 Materials

In this research, the dataset used in the study is proxy server log data from proxy devices taken based on monitoring in December 2023 in our university's network. **Figure 2** shows the Malware datasets from our Proxy Log Data.

In addition, public datasets taken from kaggle.com are used as a sample of malicious URLs for training datasets. **Figure 3** shows the sample of the public dataset used in this study.

The main aim of using this public data is as an additional data for developing the model of malicious URL classifications (Shane et al., 2023). The model obtained from the training process using the public data set is then used to detect the

malicious URLs in the university's network obtained from the proxy server log.

In the process of retrieving data from the proxy server log (Pannu et al., 2016), the WebProxy Log Catcher tool, which has a function to get proxy logs from the proxy by accessing the proxy IP address via Simple Network Management Protocol (SNMP) on port 514 is used. The obtained proxy log data will be saved to the computer drive. Then, the proxy log data is exported using ProxyLog Explorer, which functions to get the URL address generated from the proxy log.

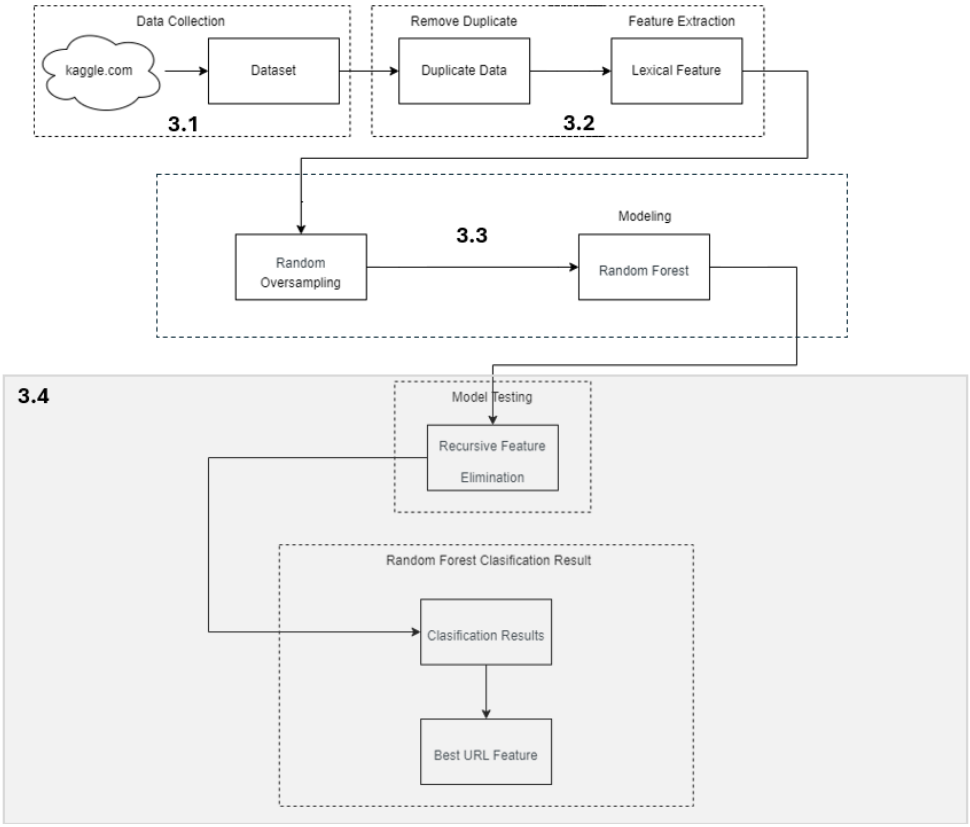
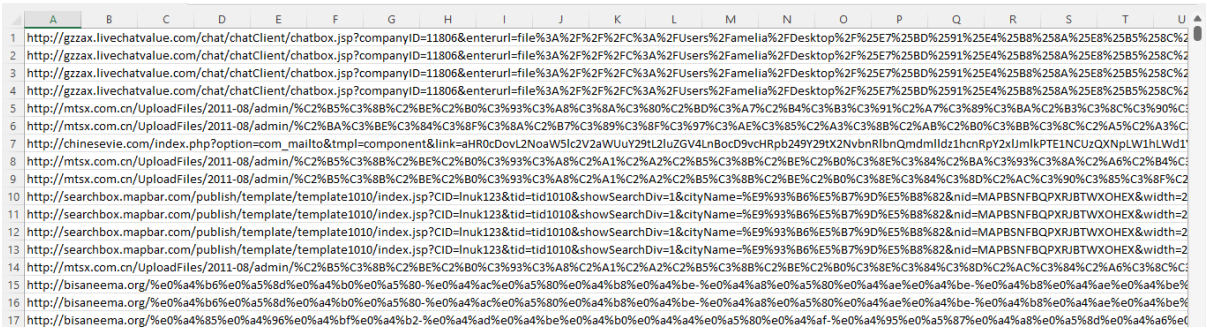


Figure 1. Process Design

	A	B	C	D	E
1	Request	Requests	Receive	Send	Visitors
2	10.31.150.1	68306			1
3	www.msftconnecttest.com/connecttest.txt	17720			97
4	Hotspot	10930			1
5	detectportal.firefox.com/canonical.html	4292			15
6	connectivitycheck.gstatic.com/generate_204	3172			193
7	(compatible;	2309			1
8	c.whatsapp.net/chat	2296			111
9	msedge.b.tlu.dl.delivery.mp.microsoft.com/filestreamingservice/files/1da2036a-93c0-4537-9	1501			1
10	conn-service-gl-01.allawntech.com/generate204	1226			19
11	dmd.metaservices.microsoft.com/metadata.svc	1184			21
12	go.microsoft.com/fwlink/?LinkID=252669&clcid=0x409	1156			21
13	files.avast.com/files/vpn_ssid_notif.txt	1145			6
14	clients3.google.com/generate_204	593			75
15	msedge.b.tlu.dl.delivery.mp.microsoft.com/filestreamingservice/files/0e0873f1-66c7-48f8-87	580			1
16	tlu.dl.delivery.mp.microsoft.com/filestreamingservice/files/48a79769-4d26-4523-9e9e-d968f	444			1
17	www.msftncsi.com/ncsi.txt	429			10
18	conn-service-us-04.allawnos.com/generate204	338			24
19	msedge.b.tlu.dl.delivery.mp.microsoft.com/filestreamingservice/files/1da2036a-93c0-4537-9	299			1
20	ping.urban-vpn.com/	296			1
21	heartbeat.dm.origin.com/	285			2

Figure 2. Proxy Log Data



	A
1	http://gzzax.livechatvalue.com/chat/chatClient/chatbox.jsp?companyID=11806&enterurl=file%3A%2F%2F%2FC%3A%2FUsers%2Famelia%2FDesktop%2F%25E7%25BD%2591%25E4%25B8%258A%25E8%25B5%258C%2
2	http://gzzax.livechatvalue.com/chat/chatClient/chatbox.jsp?companyID=11806&enterurl=file%3A%2F%2F%2FC%3A%2FUsers%2Famelia%2FDesktop%2F%25E7%25BD%2591%25E4%25B8%258A%25E8%25B5%258C%2
3	http://gzzax.livechatvalue.com/chat/chatClient/chatbox.jsp?companyID=11806&enterurl=file%3A%2F%2F%2FC%3A%2FUsers%2Famelia%2FDesktop%2F%25E7%25BD%2591%25E4%25B8%258A%25E8%25B5%258C%2
4	http://gzzax.livechatvalue.com/chat/chatClient/chatbox.jsp?companyID=11806&enterurl=file%3A%2F%2F%2FC%3A%2FUsers%2Famelia%2FDesktop%2F%25E7%25BD%2591%25E4%25B8%258A%25E8%25B5%258C%2
5	http://mtsx.com.cn/UploadFiles/2011-08/admin/%C2%B5%C3%8B%C2%BE%C2%B0%C3%93%C3%A8%C3%8A%C3%80%C2%BD%C3%A7%C2%B4%C3%B3%C3%91%C2%A7%C3%89%C3%BA%C2%B3%C3%8C%C3%90%C3
6	http://mtsx.com.cn/UploadFiles/2011-08/admin/%C2%BA%C3%BE%C3%84%C3%8F%C3%8A%C2%B7%C3%89%C3%8F%C3%97%C3%AE%C3%85%C2%A3%C3%8B%C2%AB%C2%B0%C3%BB%C3%8C%C2%A5%C2%A3%C2
7	http://chinesevie.com/index.php?option=com_mailto&tmpl=component&link=aHR0cDovL2NoaW5lc2V2aWUuY29tL2luZGV4LnBocD9vcHRpb249Y29tX2NvbnRlbnQmdmlldz1hcnRpY2xlJmlkPTE1NCUzQXNpLW1hLWd1
8	http://mtsx.com.cn/UploadFiles/2011-08/admin/%C2%B5%C3%8B%C2%BE%C2%B0%C3%93%C3%A8%C2%A1%C2%A2%C2%B5%C3%8B%C2%BE%C2%B0%C3%8E%C3%84%C2%BA%C3%93%C3%8A%C2%A6%C2%B4%C3
9	http://mtsx.com.cn/UploadFiles/2011-08/admin/%C2%B5%C3%8B%C2%BE%C2%B0%C3%93%C3%A8%C2%A1%C2%A2%C2%B5%C3%8B%C2%BE%C2%B0%C3%8E%C3%84%C3%8D%C2%AC%C3%90%C3%85%C3%8F%C2
10	http://searchbox.mapbar.com/publish/template/template1010/index.jsp?CID=lnuk123&tid=tid1010&showSearchDiv=1&cityName=%E9%93%B6%E5%B7%9D%E5%B8%82&nid=MAPBSNFBQPXRJBTWXOHEX&width=2
11	http://searchbox.mapbar.com/publish/template/template1010/index.jsp?CID=lnuk123&tid=tid1010&showSearchDiv=1&cityName=%E9%93%B6%E5%B7%9D%E5%B8%82&nid=MAPBSNFBQPXRJBTWXOHEX&width=2
12	http://searchbox.mapbar.com/publish/template/template1010/index.jsp?CID=lnuk123&tid=tid1010&showSearchDiv=1&cityName=%E9%93%B6%E5%B7%9D%E5%B8%82&nid=MAPBSNFBQPXRJBTWXOHEX&width=2
13	http://searchbox.mapbar.com/publish/template/template1010/index.jsp?CID=lnuk123&tid=tid1010&showSearchDiv=1&cityName=%E9%93%B6%E5%B7%9D%E5%B8%82&nid=MAPBSNFBQPXRJBTWXOHEX&width=2
14	http://mtsx.com.cn/UploadFiles/2011-08/admin/%C2%B5%C3%8B%C2%BE%C2%B0%C3%93%C3%A8%C2%A1%C2%A2%C2%B5%C3%8B%C2%BE%C2%B0%C3%8E%C3%84%C3%8D%C2%AC%C3%84%C2%A6%C3%8C%C3
15	http://bisaneema.org/%e0%a4%b6%e0%a5%8d%e0%a4%b0%e0%a5%80-%e0%a4%ac%e0%a5%80%e0%a4%b8%e0%a4%be-%e0%a4%a8%e0%a5%80%e0%a4%ae%e0%a4%be-%e0%a4%b8%e0%a4%ae%e0%a4%be%
16	http://bisaneema.org/%e0%a4%b6%e0%a5%8d%e0%a4%b0%e0%a5%80-%e0%a4%ac%e0%a5%80%e0%a4%b8%e0%a4%be-%e0%a4%a8%e0%a5%80%e0%a4%ae%e0%a4%be-%e0%a4%b8%e0%a4%ae%e0%a4%be%
17	http://bisaneema.org/%e0%a4%85%e0%a4%96%e0%a4%bf%e0%a4%b2-%e0%a4%ad%e0%a4%be%e0%a4%b0%e0%a4%a4%e0%a5%80%e0%a4%af-%e0%a4%95%e0%a5%87%e0%a4%a8%e0%a5%8d%e0%a4%a6%e0

Figure. 3. Malware URL Dataset

3.2 Pre-Processing

The following process removes the redundant URLs from the input CSV document. Then, it will be written back into the output CSV document. This is done by creating an empty set to retain unique URLs.

The lexical feature process is the selection of features from URLs that have the purpose of detecting the presence of malicious properties. Some of the important characteristics that emerge from this procedure consist of 1) data regarding the utilization of IP addresses, 2) identification of suspicious URLs, 3) verification of the existence of URLs in Google Search Console, and 4) the calculation of different attributes such as certain symbols or the implementation of the HTTPS protocol. Moreover, specific attributes include assessing the number of subdomains in the URL, calculating the URL length, examining elements in the URL, and recognizing dubious terms in the URL used in the study.

3.3 Classification Process

The pre-processed data will be extracted using the urllib and Neural Language Toolkit (NLTK) packages contained in Python to get the value of each URL and then convert it into numeric form so that the random forest algorithm can read it.

The output file from the feature extraction process based on the appropriate classification (benign, phishing, spam, malware, and defacement) is used as an input file into the Recursive Feature Elimination (RFE) module. Furthermore, the data is scaled to normalize the feature values. The newly selected features will be trained on the Random Forest Classifier (Khammas, 2020).

The results of the training and testing datasets are displayed with accuracy scores. A bar chart is created to visualize the important features selected as determined by the Random Forest Classifier. Confusion Matrix is designed and visualized to show the model's performance in true positive, true negative, false positive, and false negative conditions.

The classification process is conducted using the new test dataset fed into the trained model. The model's performance is analyzed by calculating how much data is predicted as malware.

Table 1 presents the Lexical Features used to evaluate URL structure and content to detect potential malware on proxy servers. These features are key inputs in the Random Forest Method for classification, helping to distinguish between benign and malicious URLs based on specific patterns and characteristics. Each feature has been categorized by its Type / Measurement, which indicates how it is quantified.

Table 1 Lexical Feature

Feature Name	Type / Measurement
IP_in_url	Presence
URL_len	Length
Domain_len	Length
Dots_in_domain	Count
Hyphens_in_Domain	Count
Underscores_in_Domain	Count
Double-slash_in_URL	Count
At(@)_in_URL	Count
Hash(#)_in_URL	Count
Semicolon(;)_in_URL	Count
And(&)_in_Url	Count
Http_in_URL	Count
Https_in_URL	Count
Numbers_ratio_in_URL	Ratio
Alphabets_in_URL	Count
Alphabets_ratio_in_URL	Ratio
Lower_case_letter_in_URL	Count
Lower_case_letters_in_URL	Ratio
Upper_case_letters_in_URL	Count
Upper_case_letters_ratio_in_URL	Ratio
Special_char_in_URL	Count
Special_char_ration_in_url	Ratio
English_words_in_URL	Count
Random_Words_in_URL	Count
Avg_english_word_len_in_URL	Length
Avg_random_words_in_URL	Length

1. Presence. Some features, such as IP_in_url, detect whether an IP address is used in place of a domain name. The presence of an IP address often indicates suspicious behavior, as legitimate domains rarely use direct IP addresses.
2. Length. Features like URL_len and Domain_len measure the entire URL and domain length, respectively. Malicious URLs often have unusually long or short lengths to disguise phishing attempts or to create random URLs for attacks.
3. Count. Many features in this category, such as Dots_in_domain, Hyphens_in_Domain, Double-slash_in_URL, and Special_char_in_URL, count the occurrences of certain characters or patterns within the URL structure. For instance, the number of dots or hyphens in a domain can indicate subdomains or obscure URLs used by attackers.
4. Ratio. Features like Numbers_ratio_in_URL and Alphabets_ratio_in_URL measure the proportion of numeric or alphabetic characters within the URL. A high ratio of numbers or a low ratio of alphabets may indicate an automatically generated malicious URL.
5. Count of Special Characters. Features like At(@)_in_URL, Hash(#)_in_URL, Semicolon(;)_in_URL, and And(&)_in_Url capture the frequency of special characters often used to create complex or obscured URLs.
6. HTTP/HTTPS Protocol Detection. The presence of Http_in_URL and Https_in_URL detects the protocol used in the URL. Some malicious actors might avoid using HTTPS to bypass security checks.
7. Upper and Lower Case Letters. Features like Lower_case_letter_in_URL and Upper_case_letters_in_URL count the number of lowercase and uppercase characters in the URL while their respective ratio features calculate their relative proportions. The use of excessive uppercase or lowercase letters can indicate suspicious patterns.

3.4 Analysis

The performance of the Random Forest model was evaluated using a variety of metrics, including accuracy, precision, recall, F1-Score, and Confusion matrix. Accuracy is the proportion of correctly predicted labels from the total predictions, as shown in Equation (1).

Precision, Recall, and F1-Score metrics were calculated to provide insight into the model's performance distinguishing different types of malicious URLs (phishing, spam, malware, etc.). Precision, Recall, and F1-Score are shown in Equation (2-4).

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ prediction} \quad (1)$$

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (2)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (3)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

Confusion Matrix was used to visualize the classification model's *true positive*, *false positive*, *true negative*, and *false negative* rates. The confusion matrix is shown in **Table 2**.

These evaluation metrics provided a comprehensive view of the model's effectiveness in detecting and classifying malicious URLs.

Furthermore, based on the data that has been obtained, the author selects relevant datasets from the proxy server log. The proxy server log has several parameters: client IP, date, response code, request method, user, request, receive, send, user agent, and server IP. URL is a parameter used in this study. Each URL is generated based on user activity using the internet, so the number of URLs obtained is very useful in research.

Table 2 Confusion Matrix

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

In this stage, duplicate url data is removed so that it displays the data as needed. Then, the previously obtained data is extracted using the lexical feature model to get the value of each feature from the URL data (Joshi et al., n.d.). The value will be compared as the best feature extraction strategy using random forest as a classification. **Figure 4** shows the result of the feature extraction process conducted in the study.

4. RESULTS AND DISCUSSION

4.1. Results

URL data has several characteristics, such as URL length, number of subdomains, number of unique characters, etc. A malicious URL can be a phishing URL where the URL resembles the login page of a website, but the purpose of the URL is to steal the user's credentials. In addition, there are URLs for spreading ransomware where the URL directs users to download a ransomware file.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	URL	Querylen	domain_t	Domain Le	path_toke	avgdomai	longdoma	avgpathtokenlen	tld	Https_in_l	charcomp	charcomp	ldl_domai	ldl_path	ldl_filena	ldl_ge
2	armmf.adobe.com/arm-manifests/win/ArmMar	0	1	0	3	0.0	0	15.666.666.666.666.600		0	15	3	0	16	16	
3	cacerts.digicert.com/DigiCertTrustedG4CodeSig	0	1	0	1	0.0	0	72.0		0	18	3	0	52	52	
4	wscdn.ml.youngjoygame.com/res_version5/905	0	1	0	5	0.0	0	16.0		0	22	5	0	28	28	
5	newds02.myoppo.com/autotime/dateandtime.:	20	1	0	2	0.0	0	20.5		0	16	4	0	18	15	
6	ctldl.windowsupdate.com/msdownload/update	16	1	0	7	0.0	0	11.142.857.142.857.100		0	24	4	0	23	21	
7	akmcdn.ml.youngjoygame.com/res_mini_patch	0	1	0	7	0.0	0	19.571.428.571.428.500		0	37	5	0	53	53	
8	www.pullcm.com/relayserver/2.0/cmdreport?t	41	1	0	3	0.0	0	12.333.333.333.333.300		0	19	4	0	14	9	
9	updates.bravesoftware.com/service/update2?c	127	1	0	2	0.0	0	19.5		0	37	5	0	25	7	
10	akmcdn.ml.youngjoygame.com/res_mini_patch	0	1	0	5	0.0	0	19.0		0	24	5	0	33	15	
11	wscdn.ml.youngjoygame.com/res_version5/905	0	1	0	5	0.0	0	20.4		0	28	5	0	50	50	
12	akmcdn.ml.youngjoygame.com/res_version5/9(	0	1	0	5	0.0	0	16.8		0	22	5	0	31	31	
13	cc.imwi.ac.id/DetailProspect/42541	0	1	0	2	0.0	0	16.0		0	9	3	0	14	5	
14	akmcdn.ml.youngjoygame.com/res_version5/9(	0	1	0	5	0.0	0	17.8		0	22	5	0	36	36	
15	akmcdn.ml.youngjoygame.com/res_version5/9(	0	1	0	5	0.0	0	18.2		0	26	5	0	38	38	
16	wscdn.ml.youngjoygame.com/res_version5/905	0	1	0	5	0.0	0	16.8		0	24	5	0	32	32	
17	msedge.b.tlu.dl.delivery.mp.microsoft.com/file	100	1	0	3	0.0	0	34.0		0	36	13	0	41	36	
18	akmcdn.ml.youngjoygame.com/res_version5/9(	0	1	0	5	0.0	0	15.2		0	20	6	0	26	21	
19	akmcdn.ml.youngjoygame.com/res_version5/9(	0	1	0	5	0.0	0	19.2		0	27	5	0	43	43	
20	ctldl.windowsupdate.com/msdownload/update	16	1	0	7	0.0	0	11.142.857.142.857.100		0	26	4	0	23	21	
21	akmcdn.ml.youngjoygame.com/res_version5/9(	0	1	0	5	0.0	0	20.0		0	28	5	0	47	47	

Figure 4 Feature Extraction Results

The recursive feature elimination process is a feature selection method that adjusts the model and eliminates the weakest feature (or features) until the specified number of features is reached. Some features will be sorted based on the model attributes coef_ or feature_importances_at this stage. It also removes some features recursively every loop and tries to eliminate collinearity and dependency that may exist in the model.

From the series of developed codes, testing was carried out six times with different numbers of features, namely 10, 15, 19, 23, 29, and 35 features, as shown in **Table 3**.

The first experiment selected the number of features. As many as 10 features contained in the URL, such as the length of the ration argument, the number of dots in the URL, and others, were evaluated. Some experimental results resulted in an accuracy of 97%, indicating that the performance of the features is good. In the second experiment, the feature selection was made to 15 features, and the accuracy result was not changed to 97%. The next stage selected 19 features and gave an accuracy result of 97%. The following experiment added 23 features and showed an accuracy of 98%. Experiments were carried out with up to 35 features with an accuracy of 97%. The accuracy results are in selecting the number of features, as many as 23, which produce an accuracy of 98%. **Table 4** shows the 23 features obtained in the study.

Based on the results of Recursive Feature Elimination, 23 features were obtained, which resulted in an accuracy of 98%, and several features that represent malicious URLs were defined. For *Domain_token_count,* the number of tokens in the domain ranges from 1 to 4 and can be categorized as suspicious. For *avgdomaintokenlen and avgpathtokenlen,* average length tokens in domains and paths vary, but very low values (even zero values) indicate a possible attempt to hide the URL structure.

In the case of *urlLen* and *pathLength,* the overall length of paths and URLs varies; some URLs have very long paths, which can be used to obscure the true intent of the URL. For *Domainlength*, domains with a length between 7 and 122 can be used to trick users. Regarding spcharUrl, URLs with more than 19 special characters are often used to trick security detection.

In the case of *NumberRate_URL*, URL number ratios vary; some URLs have the highest ratio of up to 0.716981, indicating that excessive numbers are used to trick detection. Lastly, for *Entropy_Domain,* the domain entropy ranges between 1.584963 × 1015 and 2.807355 × 1016. A high level of character diversity is indicated by high entropy; malware detection systems can use this to make URLs more challenging to identify and analyze.

The next step is using the best total number of features, which is 23, and then comparing feature selection methods is conducted. Random forest is compared with other classification methods, such as information gains and select KBest.

Table 5. shows the testing results of Recursive Feature Elimination (RFE) Based on Random Forest, **Table 6**. shows the testing results of Recursive Feature Elimination (RFE) Based on Information Gains and **Table 7**. Shows Recursive Feature Elimination (RFE) testing results Based on SelectKBest. The number of features used in the process is 23. By looking at the result, it is seen that Random Forest has the highest accuracy in the classification process.

Table 3 Testing Table

Trial No.	Lexical Feature	Number of Features	Accuracy (%)
1	ArgUrlRatio, NumberofDotsinURL, argDomainRatio, pathurlRatio, host_letter_count, avgpathtokenlen, Entropy_Domain, avgdomaintokenlen, NumberRate_URL, Entropy_DirectoryName	10	97
2	argDomainRatio, ArgUrlRatio, host_letter_count, NumberofDotsinURL, NumberRate_URL, Entropy_Domain, argPathRatio, avgpathtokenlen, avgdomaintokenlen, CharacterContinuityRate, urlLen, LongestPathTokenLenght, Entropy_DirectoryName, Directory_LetterCount, tld	15	97
3	avgDomainRatio, NumberofDotsinURL, ArgUrlRatio, Entropy_Domain, NumberRate_URL, NumberRate_Extension, Directory_LetterCount, avgpathtokenlen, avgdomaintokenlen, domain_token_count, CharacterContinuityRate, domainlength, spcharUrl, host_letter_count, domainUrlRatio, argPathRatio, urlLen, SubDirLen, Domain_LongestWordLength	19	97
4	ArgUrlRatio, domain_token_count, NumberofDotsinURL, host_letter_count, Entropy_Domain, avgdomaintokenlen, argPathRation, pathurlRatio, argDomainRatio, NumberRate_URL, domainlength, CharacterContinuityRate, avgpathtokenlen, urllen, NumberRate_Extension, Domain_LongestWordLength, Directory_LetterCount, Extension_LetterCount, spcharUrl, Entropy_DirectoryName, LongestPathTokenLenght, pathLenght, URL_DigitCount	23	98
5	ArgUrlRation, NumberDotsinURL, domain_token_count, Numberrate_URL, Entropy_Domain, avgdomaintokenlen, urlLen, host_letter_count, Directory_LetterCount, avgpathtokenlen, CharacterContinuityRate, pathDomainRatio, domainlength, domainlength, domauinUrlRatio, domainlength, domainUrlRatio, argDomainRatio, Entropy_DirectoryName, Arguments_LongestWordLength, Domain_LongestWordLength, SymbolCount_FileName, Extension_LetterCount, argPathRatio, spCharUrl, NumnerRate_Extension, SymbolCount_Domain, subDirLen, LongestPathTokenLength, Extension_DigitCount, longsomaintokenlen, pathLength	29	97
6	ArgUrlRatio, NumberofDotsinURl, Entropy_Domain, NumberRate_URL, avgdomaintokenlen,host_letter_count, avgpathtokenlen, urlLen, argDomainRatio, SymbolCount_FileName, domainlength, SymbolCount_Domain, CharacterContinuityRate, Extension_LetterCount, domainUrlRatio, LongestPathTokenLength, pathLength, argPathratio, Domain_LongestWordLength, spcharUrl, pathDomainRatio, Arguments_LongestWordLength, spCharUrl, pathDomainRatio, Arguments_LongestWordLength, Directory_LetterCount, domain_token_count, NumberRate_Extension, tld, longdomaintokenlen, pathurlRatio, subDirLen, Entropy_DirectoryName, URL_DigitCount, SymbolCount_URL, Extension_DigitCount, delimeter_path, SymbolCount_Directoryname	35	97

The features selected by Random Forest provided the most effective classification of malicious URLs. While the other methods (Information Gains and SelectKBest) also performed well, and their slightly lower accuracy (97%) suggests that they might not have captured the full complexity of the feature interactions. The results demonstrate the proposed method's effectiveness, which combines lexical features and Random Forest, for detecting and preventing malware on proxy servers.

Figure 5 displays a chart of the importance of the feature of the machine learning model, more specifically, using Recursive Feature Elimination. Some parts of the chart, such as:

a. Title

Indicates that the chart represents the importance of the features determined by the RFE method.

b. Feature

The Y-axis lists the names of the features used in the model. Each feature represents a characteristic or attribute of the data used to train the model. Examples of features in the chart are 'NumberofDotsinURL,' Entropy_Domain,' and so on.

Table 4 Lexical Malware URL Prediction Feature

domain_token_count	Avgdomaintokenlen	avgpathtokenlen	urlLen	Domainlength
3	7	13	0	122
4	5	12	0	109
3	7	13	0	122
1	0	0	0	7

pathLength	pathurlRatio	ArgUrlRatio	argDomainRatio	argPathRatio
26	0	0.7295081967213115	0.0	3423076923076920
25	0	0.7064220183486238	0.0	3.08
26	0	0.7295081967213115	0.0	3423076923076920
0	0	1.0	0.0	

NumberofDotsinURL	CharacterContinuityRate	URL_DigitCount	host_letter_count	Directory_LetterCount
0	0	0	24	23
0	0	0	22	20
0	0	0	24	23
0	0	0	0	7

Extension_LetterCount	LongestPathTokenLength	Domain_LongestWordLength	spcharUrl	NumberRate_URL
3	53	13	14	0.7169811320754716
6	45	12	19	0.08888888888888889
3	53	13	14	0.6415094339622641
7	7	0	0	0.0

NumberRate_Extension	Entropy_Domain	Entropy_DirectoryName
2	1.584.962.500.721.150	0
3	28.073.549.220.576.000	0
2	1.584.962.500.721.150	0
0	22.359.263.506.290.300	0

Table 5 Testing Results of Recursive Feature Elimination Based on Random Forest

Trial No.	Lexical Feature	Number of Features	Accuracy (%)
1	ArgUrlRatio, domain_token_count, NumberofDotsinURL, host_letter_count, Entropy_Domain, avgdomaintokenlen, argPathRation, pathurlRatio, argDomainRatio, NumberRate_URL, domainlength, CharacterContinuityRate, avgpathtokenlen, urllen, NumberRate_Extension, Domain_LongestWordLength, Directory_LetterCount, Extension_LetterCount, spcharUrl, Entropy_DirectoryName, LongestPathTokenLenght, pathLenght, URL_DigitCount	23	98

Table 6 Testing Results of Recursive Feature Elimination Based on Information Gains

Trial No.	Lexical Feature	No. of Features	Accuracy (%)
1	PathLenght, subDirLen, LongestVariableValue, PathTokenLenght, DomainTokenLen, Enthropy_AfterPath, NumberRate_AfterPath, Enthropy_Extension, Entropy_URL, avgpathtokenlen, NumberRate_Extension, Enthropy_Filename, NumberRate_FileName, CharacterContinuityRate, NumberRate_DirectoryName, NumberRate_URL, PathDomainRatio, domainUrlRatio, argDomainRatio, argPathRatio, ArgUrlRatio, Entrophy_Domain	23	97

Table 7 Testing Results of Recursive Feature Elimination Based on SelectKBest

Trial No.	Lexical Feature	Number of Features	Accuracy (%)
1	domain_token_count, tld, domainlength, pathUrlRatio, domainUrlRatio, argPathRatio, NumberofDotsinURL, NumberContinuityRate, Extension_DigitCount, host_letter_count, LongestWorldLenght, Queies_variable, spchaUrl, delimeter_path, delimeter_Count, NumberRate_AfterPath, SymbolCount_URL, SymbolCount_Domain, SymbolCount_FileName, SymbolCount_Extension, SymbolCount_AfterPath, Enthropy_AfterPath	23	97

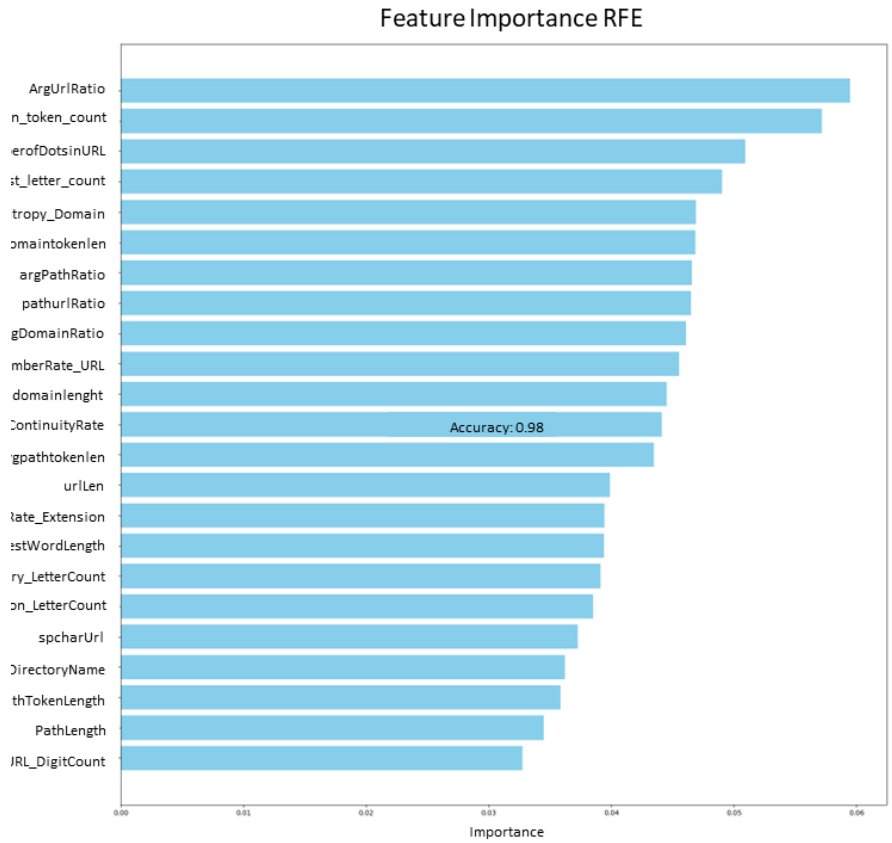


Figure 5. Results with 23 features

c. Importance

The X-axis represents the importance of each feature. The importance value shows how much a feature contributes to the model's decision-making. The higher the value, the more important the feature is.

d. Bars

Each bar corresponds to a feature and its importance. Longer bars indicate a higher level of importance.

e. Accuracy Annotation

The chart has an annotation stating, "Accuracy: 98%". This indicates that the model achieved 98% accuracy on the evaluated data. A high accuracy value indicates that the model performed well.

This chart helps identify which features are most important to model performance, providing an understanding of the aspects of the data that are most influential in predicting the target variable.

Random forest Classification results gave good results in performing URL classification after Recursive Feature Elimination (RFE) by selecting the 23 best features. Although the use of this model shows effective results, there are still some slight improvements in handling certain classes.

The next step is to test the proposed method's performance for classifying malicious URLs, where the result is shown in **Table 8**. The result indicates that the model achieves excellent performance across all categories, with particularly high precision and recall for detecting malware and spam. Defacement and benign URLs are also detected with substantial accuracy while phishing URLs show slightly lower precision but still maintain high recall. The model demonstrates its effectiveness in classifying malicious and benign URLs, making it highly suitable for real-time URL classification and malware detection.

Table 8 Table of Accuracy Result

	Precision	Recall	F1-Score	Support
Defacement	0.98	0.98	0.98	1628
Benign	0.97	0.99	0.98	1628
Malware	1.00	0.98	0.99	1628
Phishing	0.94	0.96	0.95	1628
Spam	0.99	0.98	0.99	1628
Accuracy			0.98	8140
Marco avg	0.98	0.98	0.98	8140
Weighted avg	0.98	0.98	0.98	8140

4.2. Discussion

Implementing this approach for malware detection in proxy servers involves several key considerations.

First, Random Forest is computationally lightweight compared to deep learning models, making it suitable for real-time classification. However, preprocessing steps must be optimized for large-scale deployment, including feature extraction and normalization.

Second, the model can efficiently classify URLs within proxy logs, but performance may be affected when handling high-traffic enterprise networks. Future enhancements should explore distributed computing solutions.

Third, the model processes URLs in under 9 minutes for large datasets, making it viable for near-real-time applications. However, optimizations in data ingestion and feature extraction may further reduce latency.

Lastly, attackers continuously develop obfuscation techniques to evade detection. Although lexical feature analysis provides strong indicators, integrating behavioral analysis of URLs could enhance detection resilience.

This study provides a strong foundation for implementing real-time malware detection on proxy servers. By integrating the Random Forest model with proxy log monitoring systems, IT departments can detect and prevent malware more effectively before it spreads through the network. Future work could explore additional feature engineering techniques and the integration of dynamic features from real-time network traffic to enhance detection capabilities further.

While the proposed approach has demonstrated high accuracy in classifying malicious URLs, several avenues for future research and enhancement remain. Beyond static lexical feature analysis, incorporating behavioral signals (e.g., user interaction logs and network traffic patterns) could improve malware detection accuracy, particularly for zero-day threats. In addition to that, Malicious URL structures constantly evolve to bypass traditional detection mechanisms. Future models should incorporate adaptive learning techniques to recognize emerging attack patterns. While Random Forest performed well in this study, further comparisons with advanced deep learning models (e.g., Transformers, LSTMs) may provide insights into trade-offs between computational efficiency and detection accuracy. Finally, Testing the model in large enterprise environments with real-world network traffic will validate its scalability and reliability in high-throughput proxy server scenarios

5. CONCLUSION

Applying the Random Forest algorithm and Recursive Feature Elimination (RFE) for feature selection has demonstrated highly effective results in detecting malicious URLs. By focusing on lexical features, this study successfully classified various types of malware, including phishing, spam, and defacement URLs, achieving an overall accuracy of 98%. The Random Forest model outperformed other classifiers, such as those based on Information Gains and SelectKBest, which achieved slightly lower accuracy scores of 97%. This reinforces the suitability of the Random Forest algorithm in handling a complex, multi-class classification task involving various URL categories. Performance evaluation metrics, including Precision, Recall, and F1-Score, further validated the model's robustness, particularly in identifying malware and spam with near-perfect precision and recall values. Although phishing URLs exhibited slightly lower precision, the recall remained high, ensuring the detection of most phishing threats.

.

REFERENCES

Abbas, S. H., Naser, W. A. K., & Kadhim, A. A. (2023). Subject review: Intrusion Detection System (IDS) and Intrusion Prevention System (IPS). *Global Journal of Engineering and Technology Advances, 14*(2), 155–158. https://doi.org/10.30574/gjeta.2023.14.2.0031

Ahammad, S. K. H., et al. (2022). Phishing URL detection using machine learning methods. *Advances in Engineering Software, 173*, 103288. https://doi.org/10.1016/j.advengsoft.2022.103288

Al-Janabi, M., & Altamimi, A. M. (2020). A comparative analysis of machine learning techniques for classification and detection of malware. Proceedings of the 2020 21st International Arab Conference on Information Technology (ACIT 2020). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACIT50332.2020.9300081

Alsaedi, M., Ghaleb, F. A., Saeed, F., Ahmad, J., & Alasli, M. (2024). Multi-modal features representation-based convolutional neural network model for malicious website detection. *IEEE Access, 12*, 7271–7284. https://doi.org/10.1109/ACCESS.2023.3348071

Borra, S. R., Gayathri, B., Rekha, B., Akshitha, B., & Hafeeza, B. (2023). K-nearest neighbor classifier for URL-based phishing detection mechanism.

Calderon, P., Hasegawa, H., Yamaguchi, Y., & Shimada, H. (2018). Malware detection based on HTTPS characteristics via machine learning. In *ICISSP 2018 - Proceedings of the 4th International Conference on Information Systems Security and Privacy* (pp. 410–417). SciTePress. https://doi.org/10.5220/0006654604100417

Dhingra, V., & Singh, K. P. (2023). Detecting and analyzing malware using machine learning classifiers. In H. O. Bansal, R. C. Ajmera, & S. C. Bansal (Eds.), *Next Generation Systems and Networks* (pp. 197–207). Springer Nature Singapore.

Fawagreh, K., Gaber, M. M., & Elyan, E. (2014). Random forests: From early developments to recent advancements. *Systems Science and Control Engineering, 2*(1), 602–609. https://doi.org/10.1080/21642583.2014.956265

Hemalatha, J., Roseline, S. A., Geetha, S., Kadry, S., & Damaševičius, R. (2021). An efficient

DenseNet-based deep learning model for malware detection. *Entropy, 23*(3). https://doi.org/10.3390/e23030344

Huang, Y., et al. (2023). Graph neural networks and cross-protocol analysis for detecting malicious IP addresses. *Complex and Intelligent Systems, 9*(4), 3857–3869. https://doi.org/10.1007/s40747-022-00838-y

Jasim, A. D., & Farhan, R. I. (2023). Intelligent malware classification based on network traffic and data augmentation techniques. *Indonesian Journal of Electrical Engineering and Computer Science, 30*(2), 903–908. https://doi.org/10.11591/ijeecs.v30.i2.pp903-908

Joshi, A., Lloyd, L., & Westin, P. (n.d.). Using lexical features for malicious URL detection—A machine learning approach.

Karajgar, M. D., et al. (2024). Comparison of machine learning models for identifying malicious URLs. In *2024 IEEE International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS)* (pp. 1–5). IEEE. https://doi.org/10.1109/ICITEICS61368.2024.10625423

Khammas, B. M. (2020). Ransomware detection using random forest technique. *ICT Express, 6*(4), 325–331. https://doi.org/10.1016/j.icte.2020.11.001

Khramtsova, E., Hammerschmidt, C., Lagraa, S., & State, R. (2020). Federated learning for cyber security: SOC collaboration for malicious URL detection. In *Proceedings - International Conference on Distributed Computing Systems* (pp. 1316–1321). IEEE. https://doi.org/10.1109/ICDCS47774.2020.00171

Kizza, J. M., *Texts in computer science*. Retrieved from http://www.springer.com/series/3191

Kokila, M., & Reddy K, S. (2025). Authentication, access control and scalability models in Internet of Things Security–A review. *KeAi Communications Co.* https://doi.org/10.1016/j.csa.2024.100057

Molinari, S., & Packer, J. (2023). *Malware science: A comprehensive guide to detection, analysis, and compliance.* Packt Publishing Ltd.

Pakhare, P. S., & Krishnan, C. N. N. (2021). Malicious URL detection using machine learning and ensemble modeling. In A. Pasumpon, S. M. Pandian, & I. Fernando (Eds.), *Computer Networks, Big Data and IoT* (pp. 839–850). Springer Singapore.

Pang, Y., Chen, Z., Peng, L., Ma, K., Zhao, C., & Ji, K. (2019). A signature-based assistant random oversampling method for malware detection. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)* (pp. 256–263). IEEE. https://doi.org/10.1109/TrustCom/BigDataSE.2019.00042

Pannu, M., Gill, B., Bird, R., Yang, K., & Farrel, B. (2016). Exploring proxy detection methodology. In *2016 IEEE International Conference on Cybercrime and Computer Forensic (ICCCF 2016)*. IEEE. https://doi.org/10.1109/ICCCF.2016.7740438

Pushpalatha, M., & Vijaya, A. (2023). Malicious URL website detection using selective hyper feature link stability based on soft-max deep featured convolution neural network. *International Journal on Recent and Innovation Trends in Computing and*

Communication, 11(6 S), 490–498. https://doi.org/10.17762/ijritcc.v11i6s.6957

Sharma, N. V., & Yadav, N. S. (2021). An optimal intrusion detection system using recursive feature elimination and ensemble of classifiers. *Microprocessors and Microsystems, 85*, 104293. https://doi.org/10.1016/j.micpro.2021.104293

Ujah-Ogbuagu, B. C., Akande, O. N., & Ogbuju, E. (2024). A hybrid deep learning technique for spoofing website URL detection in real-time applications. *Journal of Electrical Systems and Information Technology, 11*(1). https://doi.org/10.1186/s43067-023-00128-8