DOI: 10.24996/ijs.2025.66.6.28

ISSN: 0067-2904

Leveraging Vectorization Techniques for Malicious Website Detection With Machine Learning

Saleem Raja Abdul Samad[1*], Sundaravadivazhagan Balasubramaniyan[1], Pradeepa Ganesan [2], Chitra P [3], Poongothai K [4]

[1]IT Department, University of Technology and Applied Sciences-Shinas, Sultanate of Oman.
[2] IT Department, University of Technology and Applied Sciences-Musannah, Sultanate of Oman
[3]Department of Computer Science, School of Sciences, GITAM University, Bangalore, India.
[4]Department of Computer Science, Shri Sakthikailassh Women's College, Salem, Tamil Nadu, India

Received: 29/9/2024 Accepted: 18/11/2024 Published: 30/6/2025

Abstract

Malicious websites are those that are created to harm visitors or exploit their information for illegal purposes. These websites are commonly utilized in attacks, such as phishing, malware distribution, and scams. Clicking on a malicious URL can result in catastrophic outcomes, such as data breaches, financial losses, and identity theft. Detecting and blocking these websites is essential for protecting individuals and organizations from online threats, preserving data security, and sustaining confidence in online platforms. Researchers presented several methods for detecting malicious websites. Due to the threat's evolution, the problem remains unsolved. This paper presents a machine-learning model for malicious website identification. To process the textual contents, the experiment uses four different text vectorization techniques such as Count, TF-IDF, Hashing and word embedding (average Word2Vec). Eight machine learning models are tested to assess performance. The outcome demonstrates that the average word2vec embedding with extreme gradient boosting and random forest obtains 94.76% and 94.70% accuracy, respectively.

Keywords: Vectorizer, Malicious URL, Machine learning, Phishing, Word2Vec.

1. Introduction

In today's digitally interconnected world, the threat of malicious websites and the need to identify them cannot be overstated. These websites act as breeding grounds for different cyber threats, endangering people, companies, and even governments [1]. Understanding this threat is essential for implementing strong cybersecurity measures. Malicious websites distribute malware, ransomware, and viruses, which cause data breaches, financial losses, and system disruptions. They are crucial to phishing attempts because they deceive users into disclosing private information like login passwords and financial information [2].

By collecting personal information on these fraudulent websites, identity theft, a prevalent cybercrime, grows, endangering people's privacy and financial security. Financial fraud schemes are disseminated via fraudulent online stores and investment hoaxes hosted on malicious websites, which prey on unsuspecting victims. Cybercriminals can organize large-scale Distributed Denial of Service (DDoS) attacks, which cause broad problems, through these sites. Furthermore, ransomware attacks use fraudulent websites to encrypt victims' data and

*Email: saleem.abdulsamad@utas.edu.com

demand ransom payments for the decryption keys [3]. The harm extends beyond the victims themselves. When impersonated for fraudulent purposes by malicious websites, businesses, and organizations reputations and trustworthiness can be damaged. Some of these websites are part of a nation-state's cyber espionage campaign, which targets vital infrastructure and sensitive government systems, thereby raising national security concerns.
To deceive users into compromising their security, malicious websites use social engineering techniques to exploit human vulnerabilities [4]. By taking advantage of unpatched vulnerabilities, their drive-by downloads automatically and secretly infect victims' devices. The exploitation of zero-day vulnerabilities highlights the necessity of timely detection to counteract emergent threats [5]. Malicious websites can be hosted on cloud infrastructure, highlighting how crucial detection is to cloud security. The early detection of harmful websites helps in the proactive hunt for cyber threats, enabling cybersecurity experts to find potential security breaches before they worsen.

The detection of malicious websites is a crucial area for cybersecurity research. Researchers use a variety of strategies and approaches to find and block these risky websites. Blacklisting, heuristic rules, and machine learning models are different approaches used for detecting malicious websites. Blacklisting entails keeping a database of known harmful domains or websites [6][7]. The system checks the URL against a blacklist each time a user attempts to visit a website. If a match is found, entry is blocked, keeping users safe from known dangers. Blacklisting is useful for identifying known dangerous entities, but it may have trouble recognizing new or quickly evolving threats.

Heuristic rules are pre-programmed algorithms that alert web users to suspicious patterns or behavior on specific websites [8]. These guidelines are made to identify common traits found on malicious websites. An initial layer of defense can be quick to construct and capable of catching some frequent and visible harmful features. Heuristic rules may not be effective against sophisticated attacks and can result in false positives.
Machine learning tools analyze massive datasets using cutting-edge algorithms to identify intricate patterns linked to dangerous websites [9]. By extracting features from websites and training on labeled data, machine learning models can identify subtle nuances indicative of malicious intent. The main challenge of the machine learning model is to find features on the website that could be useful. The extracted features fail to maximize the potential of the dataset.

Most of the current research for detecting malicious websites or URLs utilizes the URL of the website. Since the URL is short and doesn't have much information, it could result in more false positives. To solve this issue, this paper focuses on the textual contents of the websites, especially from paragraph and division tags of the websites. Textual contents are processed by using four different text vectorization techniques Count, TF-IDF, and Hashing and word embedding (average word2vec). Eight machine learning models are tested to assess performance. The outcome demonstrates that the word embedding (average Word2Vec) combined with extreme gradient boosting and random forest yields an accuracy of 94.76% and 94.70%, respectively.

Contributions

- Processing textual contents of the websites especially textual data from paragraph and division tags.
- Using vectorizers to reduce feature engineering processing overhead.
- Vectorize the textual data using count, TF-IDF, and Hashing vectorizer.
- Generating word embedding by using contextualized text embedding technique average Word2Vec.

- The performance of vectorizer and word embedding methods was assessed using eight classification-based machine learning models. The outcomes are then compared.

The structure of the paper is as follows: The paper begins by outlining the proposed effort and underscoring the significance of the problem. Background details are then provided in the subsequent section. Following this, existing research works are summarized. The proposed method is described in another section, and the results of the experiment are presented subsequently. The paper concludes with a summary of its findings.

2. Background

Cybersecurity requires identifying and blocking malicious websites that spread malware, phishing, and scams. This procedure can be automated, and accuracy can be increased, by using machine learning algorithms. This paper evaluates eight machine learning algorithms with four different vectorization techniques in natural language processing. Vectorization is a crucial stage in preparing web page text for machine learning models to analyze and make predictions.

2.1 Vectorization Techniques

Vectorizers enable machine learning algorithms to process unstructured text data by transforming it into a numerical format. They convert text to numerical vectors. This research utilizes vectorization techniques in natural language processing such as Count, TF-IDF (Term Frequency-Inverse Document Frequency) Hashing vectorizer, and Average word2Vec word embedding.

- Count Vectorizer: Converts text documents into token counts. Each matrix column represents a different word.
- TF-IDF Vectorizer: It creates a matrix from a collection of written documents, where each document represents a row, and each unique word corresponds to a column. The TF-IDF score for a term is determined by combining the Term Frequency (TF) and Inverse Document Frequency (IDF). TF quantifies the frequency of a word within a specific document, while IDF assesses how widely the word is distributed across the entire document corpus.
- Hashing Vectorizer: It transforms a collection of text documents into a fixed-size numerical feature vector by mapping words or tokens to indices within the vector using a hashing function. The Hashing Vectorizer operates by tokenizing text data before applying a hash function to each token to determine its index in the feature space. The hash function compresses the hash value of the token into the desired range of indices, which typically corresponds to the dimension of the output vector. Thus, the vectors have a fixed length, making them suitable for input into machine learning algorithms.
- Average Wor2Vec Word Embedding: It extends the Word2Vec word embedding functionality by introducing a technique for dense vector representation of entire sentences or documents. Word2Vec takes an average of the word vectors in a sentence or text to determine its vector representation, rather than processing each word independently. The method consists of taking each word in a phrase, transforming it to its Word2Vec form, and then calculating the mean of these vectors. As a result, the entire sentence is represented as a single, concise vector that expresses its meaning. Average Word2Vec effectively captures sentences while maintaining the Word2Vec model's semantic relationships and context. By giving a consistent vector dimension, it facilitates the understanding of phrases and documents of various lengths by machine learning algorithms.

2.2 Machine Learning Algorithms

Machine learning models work quite well in identifying malicious websites. They play a crucial role in identifying and categorizing websites that contain potentially harmful or dangerous content, which contributes to the enhancement of web security and protects users from potential dangers. This paper uses textual web content to detect malicious websites using eight different machine learning methods: Logistic Regression (LogR), Support Vector Machine (SVM), Gaussian Naive Bayes (GNB), Decision Tree (DT), Random Forest (RF), Gradient Boosting (GB), Ada Boosting (AB), Extreme Gradient Boosting (XGBoost)

3. Related Works

Cyberthreats are becoming more sophisticated, making malicious website detection essential for cybersecurity. Researchers have developed many approaches to identify malicious websites. Table 1 presents the summary of recent research works in this domain.
Abubakr et.al [10] offer a framework for utilizing artificial neural network learning methods to identify fraudulent web pages. The dataset consists of both benign and harmful instances. The benign set was gathered through online web searches and Alexa website rankings, while the malicious set was gathered using some of the popular public databases like malwaredomainlist.com, StopBadWare, mwsl.org.cn, PhishTank, and malwareurl.com. The algorithm uses two feature groups: URL lexical features and page content features. The overall number of lexical features in the URL is 10, including the host's character count, URL length, host's number of digits, host's number of arguments, etc. There are 29 pages with content features, including HTML. JavaScript and obfuscated keyword checking. The study uses 4 machine learning algorithms and 1 deep learning algorithm. The results indicate that the artificial neural network (ANN) and DT both achieve accuracy levels of 95.12% and 96.01%, respectively.

Additionally, machine learning-focused solutions for identifying malicious websites were proposed by Lakshmanarao et al. [11]. A dataset from Kaggle that contains over 5,000 000 URLs was used for evaluations. To extract textual features from the text, three different techniques were used: count vectorizer, hashing vectorizer, and IDF vectorizer. Then, they built a phishing website detection model using four ML classifiers: LogR, K-NN, DT, and RF. The accuracy of the hash vectorizer and RF model was 97.5%.

Moreover, Saleem et al. [12] implement NLP to vectorize URLs. The experiment makes use of two datasets (D1 and D2). URL text is vectorized using three different techniques, including Count, TF-IDF, and Hashing. Experiments were conducted with the aid of machine learning and deep learning models,. DT with count vectorizer and RF with TF-IDF vectorizer both achieve 92.4% accuracy with the D1 dataset. With the D2 dataset, the DT with TF-IDF vectorizer achieved a higher accuracy of 99.5%. For dataset, D1, the ANN model achieves an accuracy of 89.6%, whereas, for dataset D2, it achieved an accuracy of 99.2%.

Saleem et al. [13] used machine learning and deep learning to classify risky websites based on web content. Data from the Kaggle dataset was incorporated into the experiments. The dataset was used to extract more than 206 features. The top-scoring features for the experiment were chosen using the selectKbest approach. According to the findings, RF and SVM both have an accuracy rate of 93%.

In addition, McGahagan et al. [14] evaluated the machine learning model to identify dangerous websites using HTML and JavaScript content on websites. Alexa and Cisco Talos Intelligence Group provided the dataset. 26 features were chosen for the experiment from over 1,865 website content features that were extracted. The experiment employs eight supervised machine learning models. The accuracy of the RF algorithm was 91.30% after feature transformation.

Pandiyan et al. [15] proposed a technique for identifying malicious websites. Linguistic URL features and Server features were used. Data from MillerSmiles and Phish Tank were used in this investigation. The outcomes demonstrate that the accuracy of the Light GBM is 85.5%.

Saleem et.al. [16] also proposed a system to detect malicious URLs by vectorizing the URLs with TF-IDF vectorizers and applying an ensemble classifier (weighted soft voting classifier). To classify harmful links, the proposed approach combines the best machine learning model (ensemble) with an appropriate weight. The data for the experiment was gathered from Kaggle (D1-Dataset), UNB, and Phistank (D2-Dataset). The results reveal that the weighted ensemble classifier for the D1 dataset obtained 91.4% accuracy and the weighted ensemble classifier for the D2 dataset reached 98.8% accuracy.
In addition, Aljabri et al. [17] demonstrated a phishing detection system using machine and deep learning. Two datasets, UCI and Kaggle, were used to derive hybrid features. Several Machine learning and Deep learning models were utilized in the experiment. As to the findings, Convolutional Neural Network (CNN) provides 90% accuracy, whereas Random Forest (RF) delivers 92% accuracy.
Also, a lightweight malicious URL detection approach was suggested by Saleem et al. [18]. It has been stated by the authors that blacklists and malicious URL detection systems that are based on reputation are unable to identify new threats. The authors presented a new method for training a machine learning model to identify malicious URLs based on lexical features such as URL length, the presence of a particular character, and subdomains. The outcomes demonstrate that the random forest algorithm has a 99% accuracy rate. Using URL, web content, and traffic features,
Furthermore, Mohamed et.al. [19] proposed a novel method to detect phishing attacks. Data was taken from Alexa, Siri, and Phish Tank. The accuracy of spotting phishing was tested with three different classifiers: Neural Network (NN), SVM, and RF. NN had a classification accuracy of 95.18%, SVM was 85.45%, and RF 78.89%.
Lastly, to detect malicious URLs, Saleem et.al. [20] used linguistic and vectorized URL features. The vectorization process makes full use of the URL's capabilities. There were six machine learning algorithms used for URL classification. According to the results, the proposed method outperforms the count vectorizer with the RF algorithm by achieving 92.49% accuracy

Table 1: Summary of existing research works

Author	Features	Accuracy	Issues
Abubakr et.al. [10]	URL Lexical and Web Content Features	DT: 95.12% ANN: 96.01%	Limited number of features. Results depend on the data set.
Lakshmanarao et.al.[11]	URLs vectorized using NLP methods	RF with hashing vectorizer: 97.5%	URLs are naturally short and contain limited information
Saleem et.al. [12]	URLs vectorized using NLP methods	D1: DT+ Count vectorizer and RF+ TF-IDF vectorizer : 92.4%. D2: DT + TF-IDF vectorizer: 99.5%	URLs are naturally short and contain limited information
Saleem et.al. [13]	Counting different tags of the websites	RF & SVM: 93%.	Limited number of features. Results depend on the data set.
McGahagan et.al. [14]	Web contents-HTML and Java Script features	RF: 91.30%	Limited number of features. Results depend on the data set.
Pandiyan et.al. [15]	URL lexical features, Server and web content features	Light GBM: 85.5%.	Limited number of features. Results depend on the data set.
Saleem et.al. [16]	URLs vectorized using TF-IDF methods	D1 dataset: Weighted ensemble classifier: 91.4% D2 dataset: Weighted ensemble classifier: 98.8%	URLs are naturally short and contain limited information.
Aljabri et al. [17]	URL and Content based features	RF:92% CNN: 90%	Limited number of features. Results depend on the data set.
Saleem et.al. [18]	URLs lexical features	RF: 99%	Limited number of features. Results depend on the data set.
Mohamed et.al. [19]	Web Content, URL, and Traffic features	NN: 95.18%, SVM: 85.45%, RF: 78.89%.	Limited number of features. Results depend on the data set.
Saleem et.al. [20]	URL lexical features and Vectorized Character level ngram using NLP techniques	RF: 92.49%	URLs are naturally short and contain limited information.

Most recent works just consider the URL as a classification method. Some experiments produce features by counting specific tags on websites. Less attention is paid to a website's content that provides a large number of classification features. This study focuses on the textual content of websites, particularly that which appears in paragraphs and division tags.

4. Proposed System (Web Analyzer)

Most systems classify malicious websites using URL, DNS, and server features. In addition to URL lexical features, certain systems use web content to detect specific tags, scripts, functions, and page length features. However, these systems have inherent limitations and did not make complete use of the dataset. Due to security issues, only the website's content analysis is being given less attention. The proposed machine learning-based approach for malicious website detection attempts to improve web security by automatically recognizing and classifying websites that contain harmful or malicious content. To accomplish this, the proposed system focuses on textual content extracted from both malicious and benign websites. The data set used in the experiment is compiled from widely used datasets, including URL dataset (ISCX-URL2016) [21] , UNB [22], and phistank [23]. The details of the dataset are shown in Tables 2 and 3.

Table 2: Dataset Details

Dataset	URLs
URL dataset (ISCX-URL2016)	Benign and Malicious (phishing, malware, and defacement URLs)
UNB	Benign and Malicious URLs
Phishtank	Malicious (Phishing) URLs

Table 3: Dataset Summary

Type	Count
Benign	5530
Malicious	5882

Our system focuses only on the textual content of the websites, particularly the textual content in the <p> and <div> tags. This data extraction from the URL dataset uses python libraries such as requests and BeautifulSoup, which are the common libraries in web scraping and web data extraction tasks. Following the completion of data extraction, data preprocessing commences. Before vectorization, preprocessing is a crucial step in NLP, as it serves to cleanse, transform, and prepare unprocessed text data for effective analysis. Algorithm 1 and Algorithm 2 describe the sequence for data extraction and preprocessing respectively.

Algorithm 1: Data Extraction

```
input: url_list
output: extracted_text
def text_Extraction(url_list):
    extracted_text=[]
    for each Url in url_list:
        resp= request.get(url)
            if response.status_code == 200:
              page_code= BeautifulSoup(resp.content)
             p_tags_list = page_code.find_all('p')
             div_tags_list = page_code.find_all('div')
 extracted_text.append([tag.text() for tag in p_tags_list] + [tag.text() for tag in div_tags_list])
            else:
            continue ()
end if
     end for
     return extracted_text
end
```

Algorithm 2: Processing

```
input: extracted_text
output: preprocessed_text
def preprocess_text(extracted_text):
    extracted_text = extracted_text.lowercase()
    extracted_text = clean_text(extracted_text)  # only considers text and numbers
    tokens = tokenize(extracted_text _text) # split words
    tokens = [word for word in tokens if word not in stop_words]
    preprocessed_text = preprocessed_text U (tokens)
    return preprocessed_text
```

Most machine learning algorithms accept input in the form of numerical feature vectors. Therefore, to work with text documents, each text document must be transformed into a numeric vector. This process is called text vectorization.

4.1 Vectorization Process

The generated text tokens from the preprocessing are transformed into a real-valued vector by using several NLP approaches, including count vectorizer, TF-IDF, and hashing vectorizer. The result of vectorization is a two-dimensional (2D) array. The vectorizer's features are set to 1000 to limit the 2D array's size.

4.2 The Word Embedding process

Word2Vec models [24], such as Continuous Bag of Words (CBOW), are employed to learn word embedding by predicting words from their context. After preprocessing the text data, the Word2Vec model is trained, adjusting word embedding as it iterates through the corpus. These dense vector representations capture both the semantic meaning and contextual information of each word. To ensure that sentence embedding have consistent lengths, the average Word2Vec function is applied. This involves summing up the word vectors and dividing them by the total number of words in the document.
Parameters for vectorizers and average word2vec word embedding are depicted in Table 4. The result of vectorization is a two-dimensional (2D) array. The resulting matrix representations are inputs for machine learning algorithms.

Table 4: Parameters for Vectorizers

Vectorizer	Parameters
Count	max_features=1000
TF-IDF	max_features=1000
Hashing	n_features=1000
Average Word2Vec	size=100, window=5, min_count=5, workers=4

Finally, the machine learning models are trained and tested with K-fold validation. K-fold cross-validation is a common method that is used to evaluate a model's performance and prevent overfitting. The dataset is divided into K roughly equal-sized subsets, or "folds," in this technique. The model is trained and evaluated K times, with a distinct fold serving as the validation set each time, while the remaining K-1 folds are utilized for training. The average of K evaluation results is the performance metric. Figure 1 depicts the process flow of the proposed system.

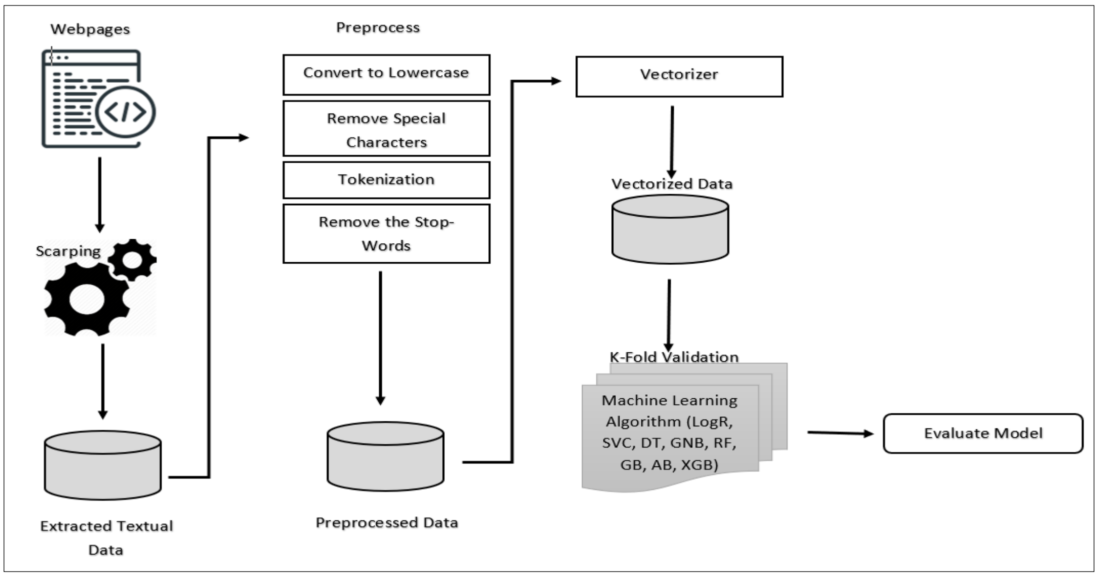

Figure 1: Proposed System (Web Analyzer)

5. Results and Discussion

The experimental environment includes windows platform, juypter notebook and python programming language with machine learning and text processing packages. The range of features produced by vectorizers is set as 1000. The outcomes of vectorizers are displayed in Tables 5–8. Table 9 shows a comparison of the best performances. The count vectorizer's performance summary is shown in Table 5. The outcome demonstrates that using random forest (RF) achieves 92.49% accuracy using 1000 features.

Table 5: Count Vectorizer Performance

Algorithm	Accuracy%	Precision%	Recall%	F1-Score%
LogR	86.41	84.22	93.04	88.09
SVC	82.79	78.92	94.60	85.60
GNB	77.88	73.09	97.59	82.90
DT	89.24	88.05	92.43	90.07
RF	92.49	94.11	91.52	92.73
GB	85.20	83.11	92.09	86.98
AB	81.86	79.73	90.85	84.28
XGB	89.40	87.83	94.00	90.55

The output of the TF-IDF vectorizer with 1000 features is displayed in Table 6. The outcome demonstrates random forest (RF) achieves an accuracy of 92.66%. The output of the hashing vectorizer with 1000 features is shown in Table 7. The outcome demonstrates that random forest (RF) attains 91.48% accuracy.

Table 6: TF-IDF Vectorizer Performance

Algorithm	Accuracy%	Precision%	Recall%	F1-Score%
LogR	88.29	87.71	91.01	89.12
SVC	91.41	91.41	92.63	91.91
GNB	78.82	73.96	97.76	83.57
DT	86.84	85.38	91.73	88.17
RF	92.66	94.13	91.83	92.90
GB	85.47	83.52	91.90	87.15
AB	85.20	83.67	90.39	86.57
XGB	90.92	89.96	93.52	91.58

Table 7: Hashing Vectorizer Performance

Algorithm	Accuracy%	Precision%	Recall%	F1-Score%
LogR	86.08	85.35	90.07	87.34
SVC	91.23	91.23	92.55	91.78
GNB	82.83	87.58	79.18	82.53
DT	85.97	84.65	90.80	87.40
RF	91.48	92.77	90.94	91.79
GB	85.23	83.05	91.78	86.90
AB	81.29	79.84	88.41	83.40
XGB	90.50	89.42	93.18	91.16

The experiment also explores the use of word2vec embedding for malicious website detection. Word2vec embedding with extreme gradient boosting algorithm achieved 94.76% accuracy and 94.70% accuracy with random forest (RF) as shown in Table 8. Figure 2 compares the efficacy of three distinct vectorizers and a word embedding using 10-fold validation.

Table 8: Performance of Average Word2Vec Word Embedding

Algorithm	Accuracy%	Precision%	Recall%	F1-Score%
LogR	88.70	88.71	89.56	89.12
SVC	90.52	90.51	91.26	90.87
GNB	78.25	86.70	68.46	76.48
DT	91.82	91.04	93.39	92.19
RF	94.70	94.85	94.92	94.88
GB	91.86	91.68	92.70	92.18
AB	88.64	88.95	89.13	89.02
XGB	94.76	94.59	95.33	94.95

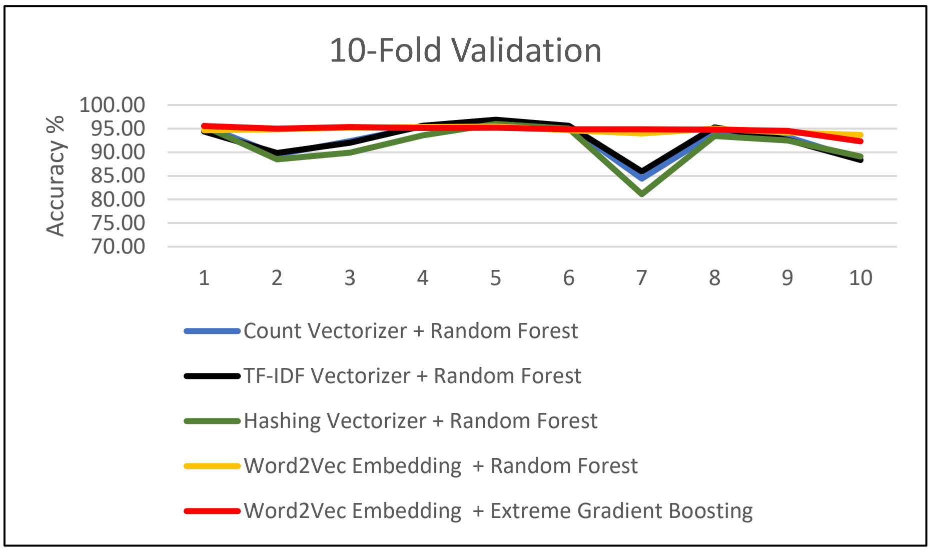

Figure 2: Performance comparison of four different Vectorizer

Table 9: Performance Comparison

Author	Accuracy
Saleem et.al. [12]	D1: DT+ Count vectorizer and RF+ TF-IDF vectorizer: 92.4%.
Saleem et.al. [13]	RF & SVM: 93%.
McGahagan et.al. [14]	RF: 91.30%
Pandiyan et.al. [15]	Light GBM: 85.5%.
Saleem et.al. [16]	D1 dataset: Weighted ensemble classifier: 91.4%.
Aljabri et al. [17]	RF:92% CNN: 90%
Mohamed et.al. [19]	NN: 95.18%, SVM: 85.45%, RF: 78.89%.
Saleem et.al. [20]	RF: 92.49%
Proposed System (Web Analyzer)	XGB: 94.76% RF: 94.70%

Existing webpage classification methods as listed in Table 9 often rely on features extracted from URL or web content. However, these features may not always provide the highest accuracy since they can lack the ability to understand the semantics and context of the content. Harmful websites can be adept at disguising themselves with similar URL patterns, and relying

solely on such features might lead to misclassifications. Effective webpage classification demands the understanding of the meaning and context of the textual content.
Malicious websites can employ sophisticated language, euphemisms, or other techniques to mask their intent. To distinguish between harmful and benign websites accurately, it is essential to grasp the nuances of language use and the context in which the content is presented. Average Word2Vec is an NLP algorithm that stands out for its ability to address these challenges. It operates by generating dense vector representations (word embedding) of words, which capture not only the meaning of individual words but also the relationships between them. Word2Vec specifically utilizes large text corpora to train neural networks to discover these embedding. This makes it superior to traditional vectorizers such as TF-IDF, Hashing, and Count Vectorizer that treat words as isolated units without considering their context. Table 9 shows that the proposed system outperforms the existing systems.

6. Conclusion

For effective and comprehensive malicious website detection, web textual content analysis is crucial. Textual elements of websites frequently conceal malicious content, such as phishing links, malicious scripts, and malware uploads. By thoroughly scrutinizing textual content displayed to users, security systems can uncover malicious patterns and identify suspicious behaviors. Additionally, analyzing the textual content enables the detection of advanced evasion techniques used by cybercriminals to avoid traditional detection methods. By leveraging the power of NLP techniques to convert website content into numerical representations and utilizing machine learning algorithms for classification, enables the automated identification of potentially harmful websites and protects users from malicious activities. The experimental findings showed that the average word2vec embedding technique is useful for classifying malicious websites. The average word2vec approach successfully captured the contextual and semantic information present in the website's textual content, enabling the classifier to distinguish between malicious and benign websites. More advanced contextualized techniques may improve detection accuracy.

7. Statements on compliance with ethical standards and standards of research involving animals

"This article does not contain any studies involving animals performed by any of the authors."

8. Disclosure and conflict of interest

"Conflict of Interest: The authors declare that they have no conflicts of interest."

References

[1] Eshete B, Villafiorita A and Weldemariam K. "Malicious Website Detection: Effectiveness and Efficiency Issues," IEEE Xplore. pp.123–126, 2011. [Online]. Available: https://doi.org/10.1109/SysSec.2011.9.

[2] National Cyber Security Centre, "Phishing attacks: defending your organization," NCSC.gov.uk 2018. [Online]. Available: https://www.ncsc.gov.uk/guidance/phishing.

[3] Kaspersky, "Ransomware Attacks and Types – How Encryption Trojans Differ," 2021. [Online]. Available: https://www.kaspersky.com/resource-center/threats/ransomware-attacks-and-types

[4] Kaspersky, "What is Social Engineering?," 2020. [Online]. Available: https://usa.kaspersky.com/resource-center/definitions/what-is-social-engineering.

[5] Kaspersky, "What is Zero Day Exploit?," 2018. [Online]. Available: https://www.kaspersky.com/resource-center/definitions/zero-day-exploit.

[6] Saleem Raja A, Madhubala R, Rajesh N, Shaheetha L and Arul N, "Survey on Malicious URL Detection Techniques," pp.778–81, 2022. [Online]. Available: https://doi.org/10.1109/ICOEI53556.2022.9777221.

[7] L Shaheetha, K. Vadivazhagan and Parvees M, "Detection of Malicious Domains in the Cyberspace using Machine Learning & Deep Learning: A Survey," pp.1540-1543, 2022. [Online]. Available: https://doi.org/ 10.1109/SMART55829.2022.10047254.

[8] Saleem Raja A, Pradeepa G and Arul N, "MUDHR: Malicious URL Detection Using Heuristic Rules based Approach," *AIP Conference Proceedings*, 2393 (1), 2022. [Online]. Available: https://doi.org/10.1063/5.0074077.

[9] Akib and Qamar, "Syed. Detecting Textual Propaganda Using Machine Learning Techniques," Baghdad Sci.J [Internet]. vol.18(1), pp.199-209, 2021. [Online]. Available: https://doi.org/10.21123/bsj.2021.18.1.0199.

[10] Abubakr S, Baharudin BB and Jung LT, "Malicious Web Page Detection: A Machine Learning Approach", Lecture Notes in Electrical Engineering. pp. 217–224, 2014. [Online]. Available: https://doi.org/10.1007/978-3-642-41674-3_32.

[11] Lakshmanarao A, Babu MR, and Bala Krishna MM, "Malicious URL Detection using NLP, Machine Learning and FLASK," *International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES),* pp.1-4, 2021. [Online]. Available: https://doi.org/10.1109/ICSES52305.2021.9633889.

[12] Saleem Raja A, Pradeepa G, Mahalakshmi S and Jayakumar MS, "Natural language based malicious domain detection using machine learning and deep learning," Scientific and Technical Journal of Information Technologies, Mechanics and Optics, vol. 23(2), pp. 304–312. 2023. [Online]. Available: https://doi.org/10.17586/2226-1494-2023-23-2-304-312

[13] Saleem Raja A, Sundarvadivazhagan B, Vijayarangan R and Veeramani S, "Malicious Webpage Classification Based on Web Content Features using Machine Learning and Deep Learning," *International Conference on Green Energy, Computing and Sustainable Technology (GECOST)*, pp. 314-319, 2022. [Online]. Available: https://doi.org/10.1109/GECOST55694.2022.10010386.

[14] McGahagan J, Bhansali D, Ciro Pinto-Coelho, and Cukier M, "A Comprehensive Evaluation of Webpage Content Features for Detecting Malicious Websites," *9th Latin-American Symposium on Dependable Computing (LADC)*, pp. 1-10, 2019. [Online]. Available: https://doi.org/10.1109/LADC48089.2019.8995713.

[15] Pandiyan S S, Selvaraj P, Burugari VK and Benadit P J and Kanmani P. "Phishing attack detection using Machine Learning. Measurement: Sensors, vol. 24, 2022. [Online]. Available: https://doi.org/10.1016/j.measen.2022.100476

[16] Saleem Raja A, Sundaravadivazhagan B, Ganesan P, Justin R and Karthikeyan R, "Weighted ensemble classifier for malicious link detection using natural language processing," International Journal of Pervasive Computing and Communications, 2023. [Online]. Available: https://doi.org/10.1108/IJPCC-09-2022-0312

[17] Aljabri M and Mirza S, "Phishing Attacks Detection using Machine Learning and Deep Learning Models," *International Conference on Data Science and Machine Learning Applications (CDMA)*, pp.175-180, 2022. [Online]. Available: https://doi.org/10.1109/CDMA54072.2022.00034.

[18] Saleem Raja A, Vinodini R and Kavitha A. "Lexical features based malicious URL detection using machine learning techniques, " Materials Today: Proceedings, vol.47(1), pp. 163-166, 2021. [Online]. Available: https://doi.org/10.1016/j.matpr.2021.04.041

[19] Mohamed G, Visumathi J, Mahdal M, Anand J and Elangovan M, "An Effective and Secure Mechanism for Phishing Attacks Using a Machine Learning Approach," Processes, vol.10(7): 1356, 2022. [Online]. Available: https://doi.org/10.3390/pr10071356.

[20] Saleem Raja A, Peerbasha S, Mohammed Iqbal Y, Sundarvadivazhagan B and Mohamed Surputheen M. "Structural Analysis of URL For Malicious URL Detection Using Machine Learning," Journal of Advanced Applied Scientific Research, vol. 5(4), pp. 28–41. [Online]. Available: https://doi.org/10.46947/joaasr542023679.

[21] Malicious URLs dataset. www.kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset?resource=download

[22] Mohammad S, Mohammad A, Arash H, Natalia S and Ali A, “Detecting Malicious URLs Using Lexical Analysis," Network and System Security, vol. 9955, pp. 467-482, 2016. [Online]. Available: https://doi.org/10.1007/978-3-319-46298-1_30

[23] PhishTank Developer Information. Cisco Talos Intelligence Group. [Online]. Available: https://www.phishtank.com/developer_info.php

[24] Noureen, Sharin Hazlin Huspi and Zafar Ali, "Sentiment Analysis on Roman Urdu Student's Feedback Using Enhanced Word Embedding Technique," Baghdad Science Journal, vol. 21 No. 2, 2024. [Online]. Available: https://doi.org/10.21123/bsj.2024.9822.