Integration of natural language processing methods and machine learning model for malicious webpage detection based on web contents

Shaheetha Liaquathali, Vadivazhagan Kadirvelu
Department of Computer and Information Science, Annamalai University, Tamil Nadu, India

Article Info

Article history:

Received Aug 1, 2024
Revised Sep 8, 2024
Accepted Oct 6, 2024

Keywords:

Cybercrime
Ensemble
Machine learning
Malicious webpage detection
Phishing
Stacking
Word2Vec embedding

ABSTRACT

Malicious actors continually exploit vulnerabilities in web systems to distribute malware, launch phishing attacks, steal sensitive information, and perpetrate various forms of cybercrime. Traditional signature-based methods for detecting malicious webpages often struggle to keep pace with the rapid evolution of malware and cyber threats. As a result, there is a growing demand for more advanced and proactive approaches that can effectively identify malicious web content based on its characteristics and behavior. Detection based on web content is crucial because malicious webpages can be designed to mimic legitimate ones, making them difficult to identify through traditional means. By analyzing the content of webpages, it becomes possible to uncover patterns, anomalies, and malicious intent that may not be evident from surface-level inspection. The proposed approach integrates a pretrained Word2Vec model with seven distinct machine learning classifiers to enhance malicious webpage detection. Initially, web contents (documents) are encoded using the Word2Vec model, followed by the computation of average Word2Vec embeddings for each document. Subsequently, each classifier is trained on the extracted average Word2Vec embedding features. The results demonstrate that the Word2Vec model significantly enhances the detection accuracy, achieving an accuracy of 94.8% and an F1-score of 94.9% with the random forest classifier, and an accuracy of 94.6% and an F1-score of 94.7% with the extreme gradient boosting classifier.

Corresponding Author:

Shaheetha Liaquathali
Department of Computer and Information Science, Annamalai University
Tamil Nadu, India
Email: shahee.aasc@gmail.com

1. INTRODUCTION

In our modern digital landscape, the internet plays an indispensable role in communication, commerce, and information sharing. However, alongside its myriad advantages, the internet also presents significant cybersecurity risks, particularly through malicious webpages [1]. These webpages harbor various forms of harmful content, including malware and phishing scams, designed to deceive users, compromise their privacy, or inflict damage on their devices [2]. Traditional methods of detecting malicious webpages often rely on URL-based blacklisting, heuristic approaches, or URL-based feature extraction. However, these methods have limitations in accurately identifying and preventing access to all types of malicious webpages [3]. URL-based blacklisting involves maintaining a database of known malicious URLs and blocking access to websites with matching URLs. While effective to some extent, this approach fails to address newly created malicious webpages or those that dynamically generate URLs. Moreover, cybercriminals can easily evade detection by frequently changing URLs or using URL obfuscation techniques. Heuristic approaches rely on predefined rules or patterns to identify potentially malicious webpages based on characteristics such as URL structure, HTML content, or behavior. While heuristic methods can detect some types of malicious activity, they often suffer from high false positive rates and may overlook sophisticated threats that do not conform to predefined patterns [4]. URL-based feature extraction involves analyzing features derived from the URL itself, such as domain reputation, URL length, or the presence of specific keywords. While this method can provide some insight into the nature of a webpage, it may not capture the full context or semantic meaning of the content hosted on the webpage [5].

The proposed approach, the web content analysis with the Word2Vec model (WCA-W2V), addresses these limitations by leveraging the actual content of webpages for malicious webpage detection. In particular, the textual contents (documents) are retrieved from paragraph, division, and meta tags embedded within webpages. These contents are then subjected to Word2Vec embedding techniques to encapsulate them into dense vector representations [6]. Subsequently, the average Word2Vec embeddings are computed for each document, enabling a more compact and semantically enriched representation of the webpage's textual content. By capturing semantic relationships between words, the approach enhances the accuracy of malicious webpage detection. Unlike URL-based methods, which focus solely on the address (URL) of the webpage, the proposed approach considers the actual content (textual) hosted on the webpage. This allows the detection of malicious content regardless of changes to URLs or evasion tactics employed by cybercriminals.

Recent research in malicious webpage detection has explored various vectorization methods in natural language processing, including count, TF-IDF, and hashing vectorizers [7]. However, the proposed approach diverges by utilizing Word2Vec embedding, which captures semantic relationships between words. Unlike count-based methods, which rely solely on word occurrences [8], Word2Vec embeddings represent words in a high-dimensional vector space, enabling a more subtle and contextually meaningful representation of web content. Similarly, while TF-IDF assigns weights based on word frequency, it lacks inherent semantic understanding [9], unlike Word2Vec embeddings which encode semantic information, thereby enhancing the accuracy of malicious content detection. Moreover, unlike hashing techniques that convert text into fixed-length hash codes [10], Word2Vec embeddings preserve semantic relationships, facilitating a deeper understanding of web content semantics and enabling the detection of subtle patterns indicative of malicious intent [11].

The major contributions of this work are i) leveraging actual webpage content for malicious webpage detection, ii) extraction of content from paragraph, division, and meta tags for analysis, iii) utilization of Average Word2Vec embedding techniques for dense and meaningful representation of webpage content, iv) enhancement of detection accuracy through capturing semantic relationships between words, v) consideration of webpage content rather than solely relying on URL-based methods, vi) ability to detect malicious content despite changes to URLs or evasion tactics, and vii) integration of machine learning classifiers with Average Word2Vec embedding for improved detection capabilities

In this paper, we aim to enhance the detection accuracy of malicious webpages utilizing advanced machine learning techniques and content analysis. Our research objective is to develop a proactive approach that effectively identifies malicious web content based on its characteristics and behavior. Specifically, we focus on integrating a pretrained Word2Vec model with machine learning classifiers to enhance malicious webpage detection. By analyzing the content of webpages, we aim to uncover patterns, anomalies, and malicious intent that may not be evident from surface-level inspection. This will contribute to the advancement of cybersecurity efforts by providing more robust and accurate methods for detecting malicious web content. The remainder of the paper is outlined as follows: Section 1 highlights the significance of the research work and elucidates its distinctions from other existing research endeavors. Section 2 delves into the existing research works within the domain of malicious webpage detection, providing a comprehensive overview of the current landscape. Section 3 furnishes background information pertinent to the research, which is essential for understanding the proposed approach. Section 4 expounds upon the proposed approach, delineating its methodology and intricacies in detail. Section 5 presents the experimental results obtained from the application of the proposed approach, showcasing its efficacy and performance metrics. Finally, Section 6 encapsulates the findings and insights derived from the research, providing concluding remarks.

2. RELATED WORKS

Detecting malicious websites is crucial for safeguarding users from online risks like malware, phishing, and data breaches. Researchers have developed various methods to identify these harmful websites. Wan *et al.* [12] outlined key characteristics of malicious websites, such as HTML and JavaScript features. Meanwhile, Malak *et al.* [13] analyzed a dataset containing 66,506 URLs to identify dangerous sites using machine learning (ML) and deep learning (DL) models. They employed three types of features—lexical, network-based, and content-based—to detect malicious websites. Techniques like correlation analysis, ANOVA, and chi-square helped extract the most significant features from the dataset. Their findings showed that the naive Bayes (NB) model achieved the highest accuracy, detecting malicious URLs with 96% precision.

Sirageldin *et al.* [14] presented a framework for identifying malicious web pages, focusing on URL keywords and page content. Their dataset consisted of both legitimate and harmful websites from sources like Alexa and MalwareURL. By splitting the features into training and testing sets, they achieved a 97% accuracy rate without any false positives. Saleem Raja *et al.* [15] also used ML and DL models to classify harmful websites based on content, particularly focusing on HTML tags, event methods, DOM keywords, and JavaScript functions. Their experiments, which used the Kaggle dataset and employed over 206 features, revealed that the support vector machine (SVM) achieved 88% accuracy, while the random forest (RF) model reached 93%.

Desai *et al.* [16] developed a Chrome extension to help users identify phishing websites, using the UCI dataset containing 11,055 records with 30 features, of which only 22 were used in the experiment. The results demonstrated that the Random Forest model achieved 96% accuracy, while SVM and k-nearest neighbor (k-NN) models achieved 93.5% and 93%, respectively. Saleem Raja *et al.* [9] proposed a lightweight algorithm for detecting malicious URLs, addressing the limitations of blacklist and reputation-based methods in detecting new threats. By utilizing lexical features such as URL length and the number of subdomains, their approach trained ML models to identify harmful URLs, with k-NN and Random Forest models achieving 98% and 99% accuracy, respectively.

Pradeepa *et al.* [17] devised a method to identify malicious URLs based on rank-based, bag-of-words, web page, and lexical features, utilizing datasets from Phistank and Kaggle. Their results showed that the random forest model offered 99% accuracy. In another study, Saleem Raja *et al.* [18] applied natural language processing (NLP) to vectorize URL terms for ML and DL model classification. They used two datasets (D1 and D2) and three vectorization techniques—count, TF-IDF, and hashing. The results indicated that the decision tree (DT) model with a count vectorizer and the RF model with a TF-IDF vectorizer both achieved 92.4% accuracy with the D1 dataset. For the D2 dataset, the DT with the TF-IDF vectorizer reached a higher accuracy of 99.5%, while the artificial neural network (ANN) model achieved 89.6% accuracy with D1 and 99.2% with D2. The comparison of various malicious URL detection methods can be seen in Table 1.

Table 1. Comparison of various malicious URL detection methods based on features and accuracy

No	Method	Features	Accuracy	Remarks
1	Malak *et al.* [13]	URL lexical features, content features, reputation features, network features	Nave Bayes: 96%	The dataset has not been fully leveraged, and the results may differ depending on the selected features.
2	Sirageldin *et al.* [14]	URL keywords and page content	97%	The features available are limited, and the results may vary based on the specific features selected for analysis.
3	Saleem Raja *et al.* [15]	HTML tags, event methods, DOM keywords, and JavaScript functions	SVM: 88% RF: 93%	Features are limited. longer processing time. unable to be extended.
4	Desai *et al.* [16]	URL lexical features, content features, reputation features, network features	RF: 96%, SVM: 93.5% k-NN 93%	Features are limited. depending on the features chosen, the outcome could vary.
5	Saleem Raja *et al.* [5]	URL lexical features, content features, reputation features, network features	k-NN: 98%, RF: 99%	The available features are constrained, and the results may fluctuate depending on which features are selected.
6	Pradeepa *et al.* [17]	Rank-based features, bag-of-words features, web page content features, URL lexical features	RF: 99%	Features are limited. Depending on the features chosen, the outcome could vary
7	Saleem Raja *et al.* [18]	URL vectorized features using NLP method	DT with TF-IDF vectorizer: 99.5%.	URLs alone may not be adequate for better classification.

Machine learning algorithms play a pivotal role in the detection of malicious webpages, contributing significantly to cybersecurity efforts [19]. Each algorithm offers unique capabilities and approaches to identifying potentially harmful web content, thereby enhancing the overall effectiveness of threat detection systems. Logistic regression, for instance, is adept at modeling the probability of a webpage being malicious based on input features. By fitting a logistic function to the data, logistic regression (LogR) provides interpretable results, allowing cybersecurity experts to understand the contributing factors to webpage threats. SVM excel in classifying webpages by separating them into different categories using hyperplanes in high-dimensional space. SVMs are particularly useful for identifying complex patterns and outliers, enhancing the detection of sophisticated malicious activities on the web.

Another valuable tool for malicious webpage detection is the similarity-based classification provided by k-NN. By examining the characteristics of neighboring webpages, k-NN can accurately classify new instances based on their proximity to known malicious or benign examples. Decision trees offer a transparent and intuitive approach to malicious webpage detection, breaking down the classification process into a series of binary decisions based on features such as URL structure, content, and behavior. Random forests (RF) extend decision trees by aggregating multiple trees and reducing overfitting, leading to improved generalization performance. Gradient boosting (GB) algorithms, including gradient boosting and extreme gradient boosting (XGBoost), sequentially build an ensemble of weak learners to iteratively correct errors, resulting in robust predictive models for identifying malicious web content.

3. METHODS

The proposed WCA-W2V for malicious webpage detection consists of several stages. First, it involves extracting and cleaning web content to ensure data quality. Next, the content is vectorized to transform it into a suitable format for analysis. Finally, the vectorized data is classified using machine learning algorithms to enhance detection accuracy, as depicted in Figure 1.

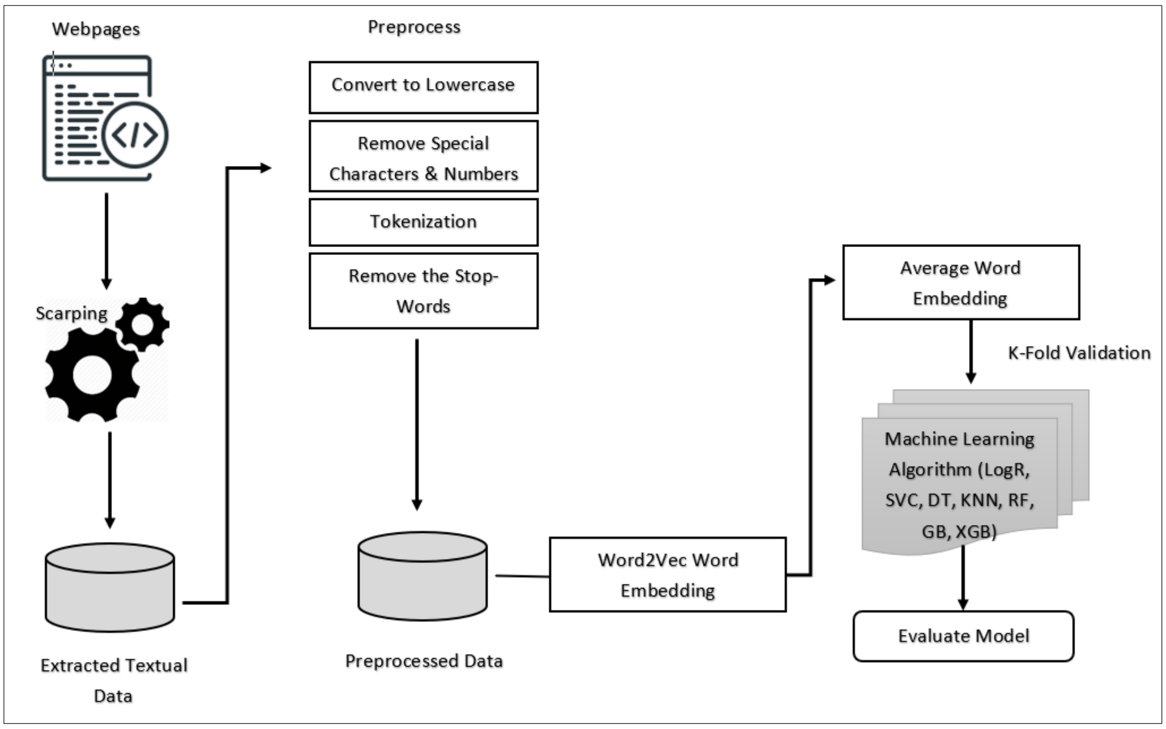

Figure 1. Proposed approach (WCA-W2V)

3.1. Dataset

The experiment incorporated renowned datasets including the URL dataset (ISCX-URL2016) [20], UNB [21], and Phistank [22]. In machine learning, imbalanced data is a common hurdle, where one class dominates with pointedly more trials than the others. This disparity can result in biased models that list the common class, leading to inferior performance concerning the minority class [10]. To address this issue and uphold impartial outcomes, an equal number of benign and malicious URLs were thoughtfully nominated for experimentation. Table 2 offers a succinct overview of the total benign and malicious URLs utilized in this study.

Table 2. Dataset summary

No	Type	Count
1	Benign	5531
2	Malicious	5883

3.2. Information extraction

During the information extraction stage, the primary aim is to retrieve textual content from the webpage, with particular attention to paragraph (para), division (div), and meta tags. This endeavor is facilitated by utilizing Python packages like requests and BeautifulSoup, which streamline the processes of web scraping and HTML parsing. Initially, the requests package is utilized to acquire the HTML content of the designated webpage through an HTTP request. Subsequently, BeautifulSoup is employed to navigate the HTML structure and extract pertinent tags containing textual data. Through iterative processing of each extracted tag, the raw textual content is systematically gathered and ready for subsequent analysis and processing. Algorithm 1 summarizes the entire process of information extraction.

Algorithm 1. Information extraction

```
Input: Set of Webpage URLs
Output: Textual content extracted from paragraph, division, and meta tags
$for\ each\ Webpage_{URL} \in Set_{of_{URL}} do$
$HTML \leftarrow Request(Webpage_{URL})$
$Tags \leftarrow Parse(HTML)$
$Text_{Content_{Temp}} \leftarrow \emptyset$
$for\ each\ Tag\ \in Tags\ do$
$if\ Tag\ is\ Para\ or\ Div\ or\ Meta\ Then$
 $Text_{Content_{Temp}} \leftarrow Text_{Content_{Temp}} \cup Extract_{Text}(Tag)$
$endif$
$return\ Text_{Content_{Temp}}$
```

3.3. Data cleaning

During the preprocessing stage for each webpage's textual content, a series of transformations are applied to prepare it for further analysis. Firstly, the text from each webpage is tokenized into individual words. Subsequently, each token is converted to lowercase, and any non-alphabetical tokens are removed. Finally, a filtering process eliminates any remaining tokens that belong to a predefined set of stop words, resulting in a clean and standardized representation of the textual content of each webpage. Algorithm 2 outlines the complete procedure for data cleansing.

Algorithm 2. Data cleaning

```
Input: Textual content of each webpage
Output: Cleaned textual content of each webpage
$for\ each\ webpage\ content\ C\ do$
    • $tokens_c \leftarrow \emptyset$
    • $for\ each\ token\ \in Tokenize\ (C) do$
            o $if\ Lowercase\ (token) is\ not\ a\ stop\ word\ then$
                    ▪ $tokens_c \leftarrow tokens_c\ \cup \{\ Lowercase\ (token)\ \}$
$Return\ tokens_c$
```

3.3. Average Word2Vec embedding

Following the cleansing of web contents, the subsequent stage of processing enables the representation of this content in a structured and meaningful manner. Through the application of Word2Vec embedding, individual words within the textual content are converted into dense vector representations, effectively capturing semantic relationships between words. This transformative process not only preserves the contextual intricacies inherent in the textual data but also facilitates computational analysis by encoding semantic information into numerical vectors.

Furthermore, to refine the representation of the webpage's textual content, the derived Word2Vec embeddings undergo aggregation through the computation of average Word2Vec embeddings for each document. By averaging the Word2Vec vectors corresponding to the words within each document, a condensed and semantically enriched representation of the webpage's content is achieved. This averaged embedding encapsulates the semantic essence of the document, fostering a comprehensive understanding of its textual content. The complete process is outlined in Algorithm 3.

3.4. Training and testing

Following the completion of the average word embedding stage, the subsequent phase involves training and testing the data using repeated stratified k-fold cross-validation. This methodology ensures robustness in model evaluation by repeatedly splitting the dataset into k subsets while maintaining class balance. Each fold is utilized as both training and testing data iteratively, allowing for a comprehensive assessment of the model's performance across multiple iterations. By employing this approach, the risk of overfitting is mitigated, and the generalization capability of the model is enhanced, thus enabling more reliable detection of malicious webpages [23]–[25]. In the final stage of the process, the focus shifts towards applying seven distinct machine learning algorithms—logistic regression, SVM, k-NN, decision tree, random forest, gradient boosting, and XGBoost. These algorithms are selected for their diverse approaches to classification and their potential to capture different aspects of the data. By measuring various performance metrics such as accuracy, precision, recall, and F1-score across these algorithms, a comprehensive evaluation of their effectiveness in malicious webpage detection is achieved.

Algorithm 3. Average Word2Vec
```
Input: Cleaned textual content of webpages (Text)
Output: Average Word2Vec embeddings for each document
        1. Word2Vec_Embeddings[] ← []
        2. Avg_Word2Vec_Embeddings[] ← []
        3. for each word w in Text:
            •   Word2Vec_Embeddings[w] ← Word2Vec(w)
        4. for each document d in Text:
            •   Sum_Word2Vec_Embeddings ← 0
            •   for each word w in d:
                o   Sum_Word2Vec_Embeddings +← Word2Vec_Embeddings[w]
                o   Avg_Word2Vec_Embeddings[d] ← Sum_Word2Vec_Embeddings / |d|
        5. return Avg_Word2Vec_Embeddings
```

4. RESULTS AND DISCUSSION

The experimental results were conducted with a system configured with a Windows 15 operating system running on a 2 GHz processor. The experimentation environment was facilitated through Jupyter Notebook, providing an interactive platform for coding and analysis. The programming tasks were executed utilizing the scikit-learn package, a comprehensive library for machine learning tasks in Python. Leveraging this setup, the performance of various machine learning algorithms was assessed for malicious webpage detection. The evaluation process involved testing the textual contents extracted from individual HTML tags such as paragraph (Para), meta, and division (Div), as well as combined text from all tags. The results of these experiments are presented in Tables 3 to 6, each showcasing the performance metrics of various machine learning algorithms including accuracy, precision, recall, and F1-score. A rigorous evaluation was conducted using repeated stratified k-fold cross-validation, ensuring robustness and reliability in assessing the effectiveness of each algorithm for malicious webpage detection. The assessment of the random forest algorithm is done through confusion matrices and ROC-AUC curves for various HTML tags, including paragraph as shown in Figures 2 and 3, Div as shown in Figures 4 and 5, and Meta as shown in Figures 6 and 7. Additionally, the combined textual contents' performance is illustrated in Figures 8 and 9.

Table 3. Performance of textual content of paragraph Tag

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Specificity (%)
LogR	84.5	85.6	84.7	85.1	81.8
SVC	86.5	88.1	86.4	87.3	83.5
k-NN	90.1	94.1	93.7	93.9	88.2
DT	91.4	99.3	99.8	99.5	89.6
RF	93.9	99.2	99.9	99.5	91.2
GB	89.5	92.3	90.7	91.5	87.1
XGB	93.9	99.2	99.8	99.5	91.4

Table 4. Performance of textual content of Div Tag

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Specificity (%)
LogR	83.2	83.1	84.6	83.8	80.9
SVC	85.8	85.6	88.2	86.9	82.7
k-NN	89.9	93.1	94.3	93.7	88.4
DT	91.8	98.6	99.4	99.0	89.2
RF	93.7	98.6	99.4	99.0	91.5
GB	88.8	90.5	91.6	91.1	86.3
XGB	93.5	98.5	99.4	99.0	90.7

Table 5. Performance of textual content of Meta Tag

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Specificity (%)
LogR	76.9	83.6	68.2	75.1	72.4
SVC	78.3	87.4	68.2	76.6	74.1
k-NN	87.1	90.7	91.8	91.2	85.6
DT	89.7	97.7	99.7	98.7	90.3
RF	92.4	97.6	99.8	98.7	91.7
GB	86.5	89.1	87.8	88.5	84.6
XGB	91.9	97.6	99.5	98.5	90.8

Table 6. Performance of Combined (Para, Div and Meta tags) textual contents

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Specificity (%)
LogR	87.2	87.6	87.7	87.6	85.0
SVC	89.6	90.0	89.8	89.9	87.5
k-NN	91.7	92.5	91.2	91.9	89.7
DT	92.1	91.3	93.6	92.4	90.5
RF	94.7	94.9	94.7	94.8	92.3
GB	91.1	91.0	91.7	91.4	89.8
XGB	94.6	94.4	95.1	94.7	92.0

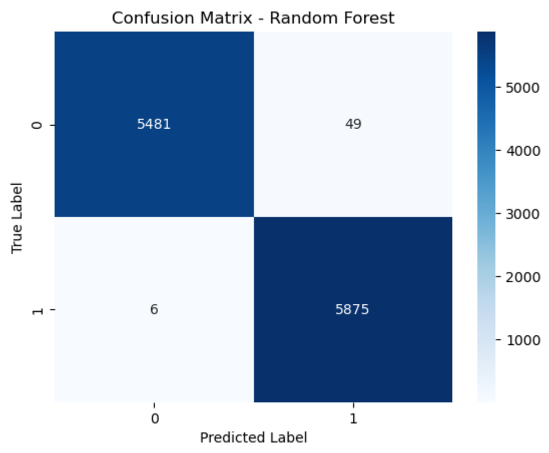

Figure 2. Confusion matrix of random forest algorithm for paragraph tag

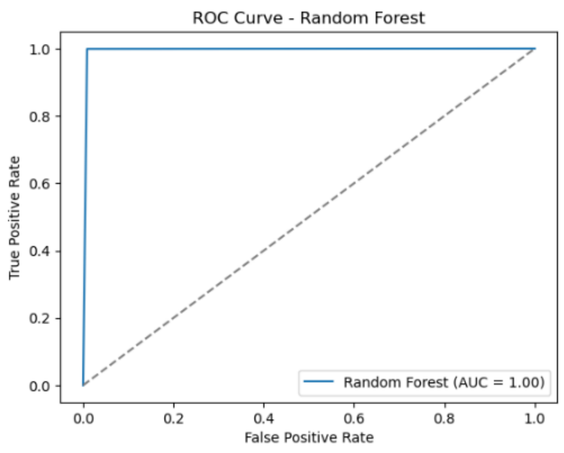

Figure 3. ROC-AUC of random forest algorithm for paragraph tag

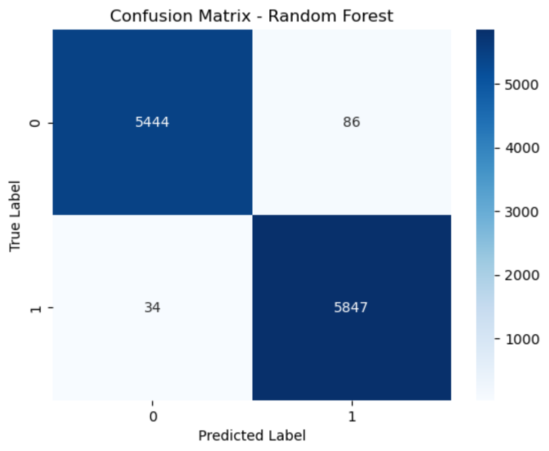

Figure 4. Confusion matrix of random forest algorithm for Div Tag

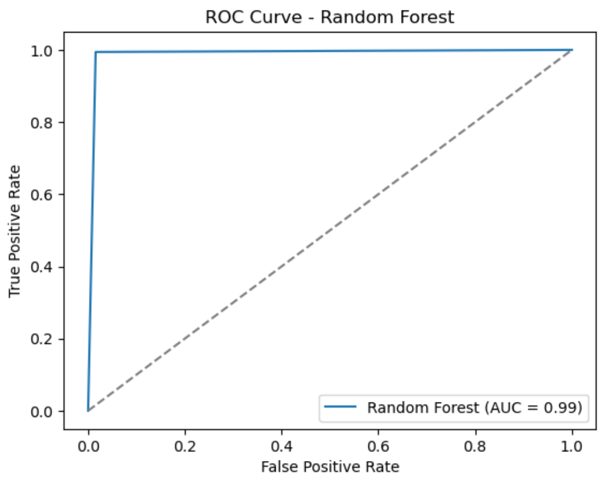

Figure 5. ROC-AUC of random forest algorithm for Div Tag

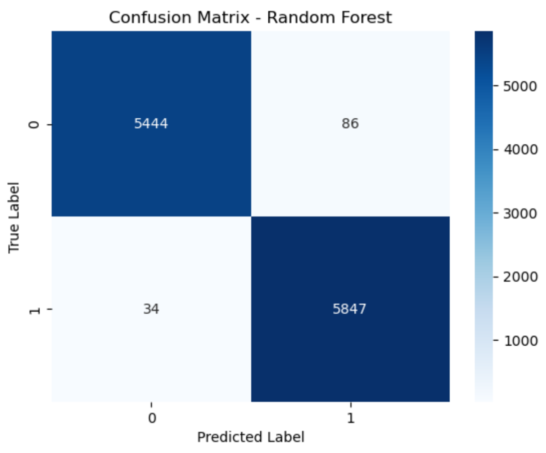

Figure 6. Confusion matrix of random forest algorithm for meta tag

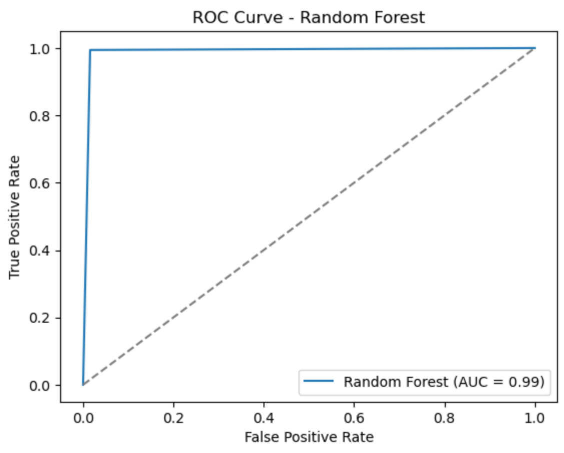

Figure 7. ROC-AUC of random forest algorithm for meta tag

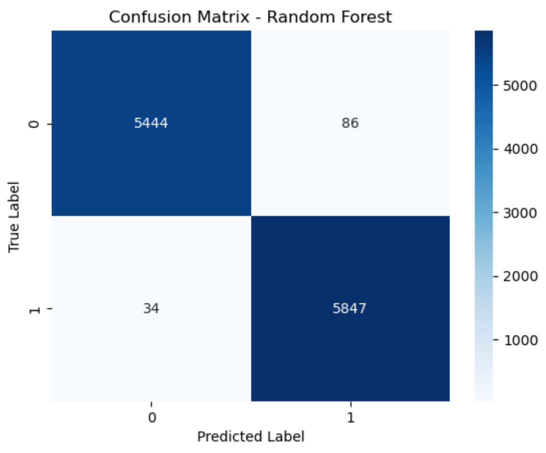

Figure 8. Confusion matrix of random forest algorithm for combined textual contents

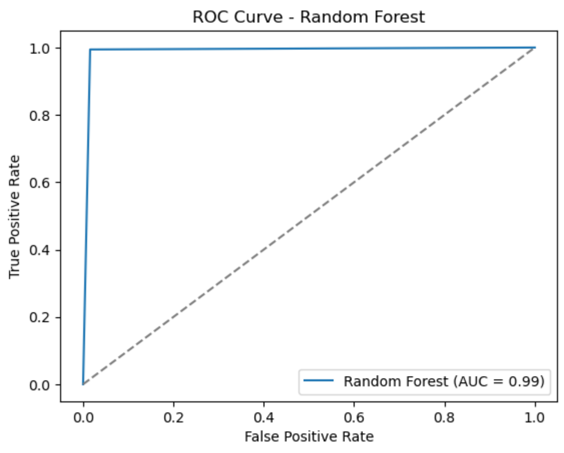

Figure 9. ROC-AUC of random forest algorithm for combined textual contents

Table 3 represents the performance metrics of various machine learning algorithms when applied to the textual content extracted specifically from the Paragraph tag. Each algorithm's accuracy, precision, recall, F1-score, and specificity are provided. Looking at the results, it's evident that all algorithms perform reasonably well across the board. RF and XGB classifiers stand out with the highest accuracy, recall, precision and F1-score among all algorithms. Specifically, the RF classifier achieves an impressive accuracy of 93.9% and an F1-score of 99.5%, closely followed by XGB with an accuracy of 93.9% and an F1-score of 99.5% as well. These algorithms demonstrate robust performance in accurately classifying malicious webpages based on the textual content from Paragraph tags. Moreover, the specificity values indicate the ability of the classifiers to correctly identify benign webpages as benign. In this context, RF and XGB classifiers maintain high specificity values, indicating their capability to effectively differentiate benign webpages from malicious ones, contributing to the overall reliability of the detection system.

Table 4 presents the performance metrics of various machine learning algorithms when applied to the textual content specifically extracted from the Div tag. Each algorithm's accuracy, precision, recall, F1-score, and specificity are listed. Observing the results, it's notable that all algorithms demonstrate commendable performance across the metrics. RF and Extreme Gradient Boosting (XGB) classifiers exhibit notably high accuracy, precision, recall, and F1-score among the algorithms assessed. Specifically, the RF classifier achieves an accuracy of 93.7% and an F1-score of 99.0%, while XGB closely follows with an accuracy of 93.5% and an F1-score of 99.0% as well. These algorithms demonstrate robust performance in accurately classifying malicious webpages based on the textual content extracted from Div tags. Moreover, the specificity values indicate the classifiers' ability to correctly identify benign webpages as benign, further enhancing the reliability of the detection system. RF and XGB classifiers maintain high specificity values, reflecting their effectiveness in distinguishing benign webpages from malicious ones.

Table 5 presents the evaluation of various machine learning systems when applied to the textual content specifically extracted from the Meta tag. Each algorithm's accuracy, precision, recall, F1-score, and specificity are listed. Upon examination of the results, it's evident that all algorithms exhibit commendable performance across the metrics. Notably, RF and extreme gradient boosting (XGB) classifiers demonstrate high accuracy, precision, recall, and F1-score among the assessed algorithms. Specifically, the RF classifier achieves an accuracy of 92.4% and an F1-score of 98.7%, while XGB closely follows with an accuracy of 91.9% and an F1-score of 98.5%. These algorithms showcase robust performance in accurately classifying malicious webpages based on the textual content extracted from Meta tags. Furthermore, the specificity values provide insights into the classifiers' ability to correctly identify benign webpages as benign. RF and XGB classifiers maintain high specificity values, indicating their effectiveness in distinguishing benign webpages from malicious ones, which contributes to the overall reliability of the detection system.

Table 6 presents the evaluation of various machine learning systems when utilizing combined textual contents extracted from paragraph (Para), division (Div), and meta tags for malicious webpage detection. The algorithms evaluated include logistic regression (LogR), support vector classifier (SVC), k-kNN, DT, RF, GB, and XGB. Across the board, all algorithms demonstrate strong performance in terms of accuracy, precision, recall, and F1-score, indicating their effectiveness in detecting malicious webpages based on combined textual content. Particularly noteworthy is the performance of RF and Extreme Gradient Boosting (XGB) classifiers, which achieve the highest accuracy of 94.7% and 94.6%, respectively, along with impressive F1-scores of 94.8% and 94.7%, respectively. These classifiers also maintain high precision and recall values, reflecting their robustness in correctly identifying both malicious and benign instances. Furthermore, the specificity metric provides insights into the classifiers' ability to accurately identify benign instances as benign. RF and XGB classifiers exhibit the highest specificity values of 92.3% and 92.0%, respectively, underscoring their effectiveness in distinguishing between benign and malicious webpages based on combined textual content. The proposed approach demonstrates superior performance compared to existing methods for content analysis of webpages. Through a comparative analysis, it has been established that our approach outperforms traditional methods in accurately detecting and analyzing malicious content embedded within webpages. Additionally, the scalability of our proposed approach surpasses that of existing methods as shown in Figure 10.

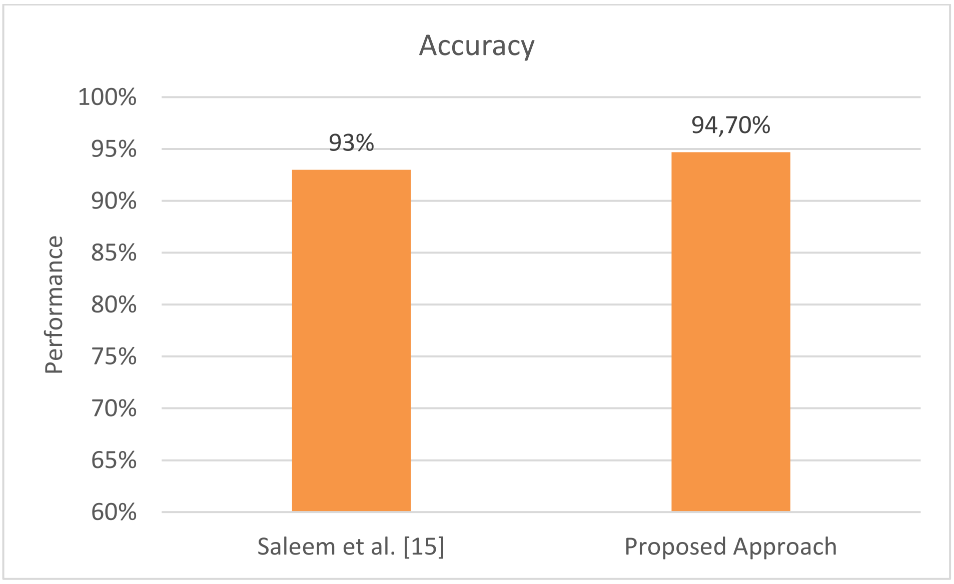

Figure 10. Performance comparison

5. CONCLUSION

The ever-evolving landscape of cyber threats underscores the critical need for robust methods to detect malicious web content. Traditional signature-based approaches often lag in identifying sophisticated malware and phishing attacks, necessitating more proactive and advanced detection techniques. Detection based on web content is particularly vital as malicious actors adeptly camouflage their activities within seemingly legitimate webpages. By delving into the underlying content, it becomes feasible to unveil subtle patterns and anomalies indicative of malicious intent. The proposed approach leverages a pretrained Word2Vec model in conjunction with seven diverse machine learning classifiers to bolster malicious webpage detection. Initially, web contents are encoded into dense vector representations using Word2Vec embeddings, followed by the computation of average embeddings for each document. Subsequent training of classifiers on these embeddings yields promising results, with the random forest classifier achieving an accuracy of 94.8% and an F1-score of 94.9%, and the extreme gradient boosting classifier attaining an accuracy of 94.6% and an F1-score of 94.7%. The proposed approach can be enhanced by incorporating more advanced context-aware pretrained models to improve detection accuracy.

REFERENCES

[1] J. M. Borky and T. H. Bradley, "Protecting information with cybersecurity," in *Effective Model-Based Systems Engineering*, Springer International Publishing, 2019, pp. 345–404.

[2] A. Saleem Raja, S. Peerbashab, Y. Mohammed Iqbal, B. Sundarvadivazhagan, and M. Mohamed 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, Jul. 2023, doi: 10.46947/joaasr542023679.

[3] C. Do Xuan, H. D. Nguyen, and T. V. Nikolaevich, "Malicious URL detection based on machine learning," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 1, pp. 148–153, 2020, doi: 10.14569/ijacsa.2020.0110119.

[4] A. S. Raja, G. Pradeepa, and N. Arulkumar, "Mudhr: malicious URL detection using heuristic rules based approach," in *AIP Conference Proceedings*, 2022, vol. 2393, doi: 10.1063/5.0074077.

[5] A. Saleem 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.

[6] Mimi, "Word2Vec for word embeddings - a beginner's guide." https://www.analyticsvidhya.com/blog/2021/07/word2vec-for-word-embeddings-a-beginners-guide/ (accessed Apr. 02, 2024).

[7] S. R. A. Samad, P. Ganesan, J. Rajasekaran, M. Radhakrishnan, H. Ammaippan, and V. Ramamurthy, "SmishGuard: leveraging machine learning and natural language processing for smishing detection," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 11, pp. 586–593, 2023, doi: 10.14569/IJACSA.2023.0141160.

[8] P. Jain, "Basics of CountVectorizer," *Towards Data Science*, 2021. https://towardsdatascience.com/basics-of-countvectorizer-e26677900f9c.

[9] Kenan Ekici, "Your TFIDF features are garbage. Here's how to fix it," *Medium*, 20222. https://medium.com/@kenanekici/your-tfidf-features-are-garbage-heres-how-to-fix-it-ca548883d8a0 (accessed Apr. 02, 2024).

[10] Kavita Ganesan, "HashingVectorizer vs. CountVectorizer." https://kavita-ganesan.com/hashingvectorizer-vs-countvectorizer/ (accessed Apr. 02, 2024).

[11] M. Suri, "A Dummy's Guide to Word2Vec," *Medium*, 2022. https://medium.com/@manansuri/a-dummys-guide-to-word2vec-456444f3c673 (accessed Apr. 02, 2024).

[12] W. N. W. Manan, A. G. A. Ahmed, and M. N. M. Kahar, "Characterizing current features of malicious threats on websites," in *Advances in Intelligent Systems and Computing*, vol. 866, Springer International Publishing, 2019, pp. 210–218.

[13] M. Aljabri *et al.*, "An assessment of lexical, network, and content-based features for detecting malicious URLs using machine learning and deep learning models," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–14, Aug. 2022, doi: 10.1155/2022/3241216.

[14] A. Sirageldin, B. B. Baharudin, and L. T. Jung, "Malicious web page detection: a machine learning approach," in *Lecture Notes in Electrical Engineering*, vol. 279, Springer Berlin Heidelberg, 2014, pp. 217–224.

[15] A. Saleem Raja, B. Sundarvadivazhagan, R. Vijayarangan, and S. Veeramani, "Malicious webpage classification based on web content features using machine learning and deep learning," in *2022 International Conference on Green Energy, Computing and Sustainable Technology, GECOST 2022*, Oct. 2022, pp. 314–319, doi: 10.1109/GECOST55694.2022.10010386.

[16] A. Desai, J. Jatakia, R. Naik, and N. Raul, "Malicious web content detection using machine leaning," in *RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings*, May 2017, pp. 1432–1436, doi: 10.1109/RTEICT.2017.8256834.

[17] G. Pradeepa and R. Devi, "Lightweight approach for malicious domain detection using machine learning," *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, vol. 22, no. 2, pp. 262–268, Apr. 2022, doi: 10.17586/2226-1494-2022-22-2-262-268.

[18] A. S. Saleem Raja, G. Pradeepa, S. Mahalakshmi, and M. S. Jayakumar, "Natural language based malicious domain detection using machine learning and deep learning," *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, vol. 23, no. 2, pp. 304–312, Apr. 2023, doi: 10.17586/2226-1494-2023-23-2-304-312.

[19] S. R. Abdul Samad *et al.*, "Analysis of the performance impact of fine-tuned machine learning model for phishing URL detection," *Electronics (Switzerland)*, vol. 12, no. 7, Mar. 2023, doi: 10.3390/electronics12071642.

[20] M. Siddhartha, "Malicious URLs dataset," *Kaggle*, 2021. https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset?resource=download (accessed Apr. 02, 2024).

[21] "URL dataset (ISCX-URL2016)," *Canadian Institute for Cybersecurity*, 2016. https://www.unb.ca/cic/datasets/url-2016.html (accessed Apr. 02, 2024).

[22] "'PhishTank,' [Online]. Available: https://www.phishtank.com/developer_info.php. [Accessed: 2 Apr 2024]."

[23] A. B. F. Khan, K. Kamalakannan, and N. S. S. Ahmed, "Integrating machine learning and stochastic pattern analysis for the forecasting of time-series data," *SN Computer Science*, vol. 4, no. 5, Jun. 2023, doi: 10.1007/s42979-023-01981-0.

[24] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, vol. 3, pp. 111–132, 2022, doi: 10.1016/j.aiopen.2022.10.001.

[25] T. Wu, M. Wang, Y. Xi, and Z. Zhao, "Malicious URL detection model based on bidirectional gated recurrent unit and attention mechanism," *Applied Sciences (Switzerland)*, vol. 12, no. 23, Dec. 2022, doi: 10.3390/app122312367.

BIOGRAPHIES OF AUTHORS

Shaheetha Liaquathali is a research scholar in the Department of Computer and Information Science at Annamalai University, Tamil Nadu, India. She has published her work in various national and international journals and conferences. Her research interests include machine learning, data mining, and cybersecurity. She can be reached at shahee.aasc@gmail.com.

Vadivazhagan Kadirvelu is an assistant professor in the Department of Computer and Information Science at Annamalai University, Tamil Nadu, India. He specializes in cybersecurity, machine learning, and data analytics. With a strong academic background and extensive research experience, Vadivazhagan has published numerous papers in national and international journals and conferences. His work focuses on the development of innovative algorithms and frameworks for enhancing data security and analytics. He is dedicated to advancing knowledge in his field and contributing to the academic community through teaching, research, and collaboration. He can be reached at vadivazhagan.k@gmail.com.