Indonesian Journal of Electrical Engineering and Computer Science
Vol. 37, No. 1, January 2025, pp. 232~240
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v37.i1.pp232-240

A novel model for detecting web defacement attacks transformer using plain text features

Xuan Dau Hoang[1], Trong Hung Nguyen[2], Hoang Duy Pham[1]
[1]Faculty of Information Security, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam
[2]Faculty of Information Technology and Security, Academy of People's Security, Hanoi, Vietnam

Article Info

Article history:

Received Jul 9, 2024
Revised Aug 28, 2024
Accepted Sep 2, 2024

Keywords:

Website defacement attacks
Website defacement detection
Website defacement detection based on deep learning
Website defacement detection based on machine learning

ABSTRACT

Over the last decade, web defacements and other types of web attacks have been considered serious security threats to web-based services and systems of many enterprises and organizations. A website defacement attack can bring severe repercussions to the website owner, such as immediate discontinuance of the website operations and damage to the owner's reputation, which may lead to enormous monetary losses. Several solutions and tools for monitoring and detecting web defacements have been designed and developed. Some solutions and tools are limited to static web pages, while others can handle dynamic ones but demand significant computational power. The existing proposals' other issues are relatively low detection rates and high false alarm rates because many crucial elements of web pages, including embedded code and images are not properly processed. This paper proposes a novel model for detecting web defacements to address these issues. The model is based on the bidirectional long-short term memory (Bi-LSTM) deep learning method using features of the plain text content extracted from web pages. Comprehensive testing on over 96,000 web pages dataset demonstrates that the proposed Bi-LSTM-based web defacement detection model outperforms earlier methods, achieving a 96.04% overall accuracy and a 2.03% false positive rate.

Corresponding Author:

Xuan Dau Hoang
Faculty of Information Security, Posts and Telecommunications Institute of Technology
96A Tran Phu Rd., Ha Dong District, Hanoi, Vietnam
Email: dauhx@ptit.edu.vn

1. INTRODUCTION

Defacement attacks on web applications and websites are a form of web attack that aims to change the web page content and thereby modify their appearance [1], [2]. According to the statistical figures of Zone-h.org [3], there are approximately 500,000 defaced websites in 2020 globally and the number of defaced websites in the first 5 months of 2021 is 200,000. Figure 1 shows the screenshot of the webpage of www.sohuutritue.gov.vn, which was defaced on 30th April 2021 by the "Overthink1877" hacking group.

Many common reasons have been pointed out why web applications and websites were defaced. However, the primary reason is serious security weaknesses that exist in web applications and websites, or their background servers, which enable attackers to perform web disfigurement attacks [1], [2]. The most popular and critical weaknesses in web applications and websites are cross-site scripting (XSS), structured Query language injection (SQLi), the inclusion of remote or local files, inappropriate management of accounts and passwords, and not-updated software.

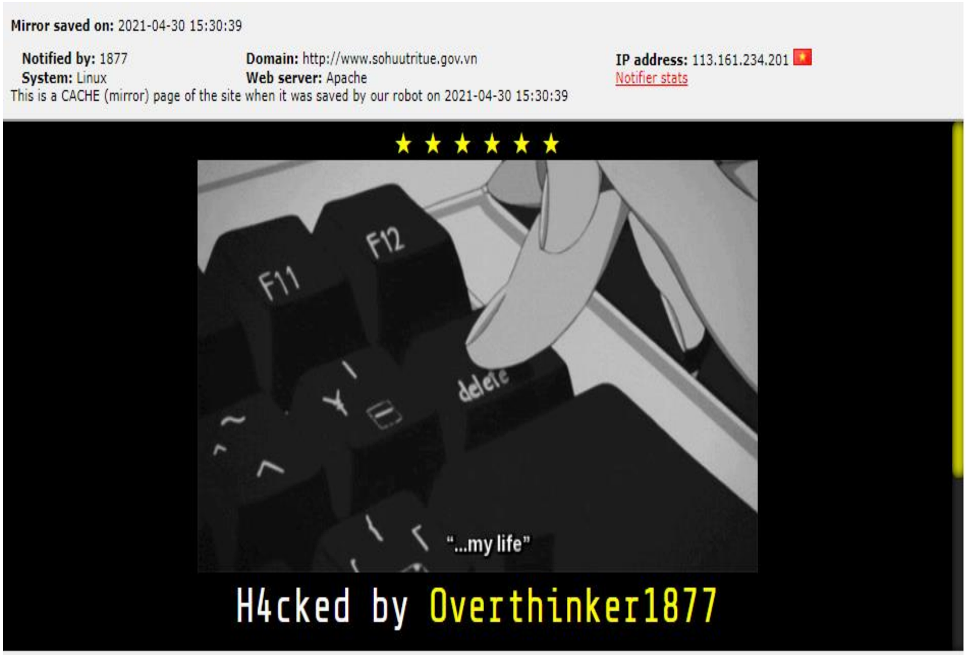

Figure 1. Screenshot of www.sohuutritue.gov.vn defaced on 30th April, 2021

A website defacement attack can result in serious effects on the website owner. The web defacement can instantly suspend normal website operations, harm the owner's renown and bring in potential losses of important data. These issues in turn may lead to large monetary losses. Since the widespread and serious repercussions of defacements on web applications and websites, many tools and solutions have been proposed and developed to combat these attacks. Current counter-measures to website defacements can be split up into 3 groups: group (1) includes tools and solutions to scan and amend security weaknesses in websites, web applications and hosting servers, including Acunetix web application security testing [4], trustwave app scanner [5] and mister vulnerability scanner [6]; group (2) consists of tools and platforms for monitoring and detecting common web attacks, including website monitoring service [7], website application monitoring [8], website defacement monitoring [9], and web orion monitor [10]; and group (3) compose of measures to detect web defacements. Common measures in group (3) will be reviewed in section 2.

This paper proposes a novel model for detecting web defacements based on deep learning techniques utilizing features of plain text content extracted from web pages to enhance the overall detection accuracy and lower the number of false alarms. The proposed method can work effectively on static web pages as well as dynamic web pages. In this detection method, the plain text features are obtained from web pages and the bidirectional long-short term memory (Bi-LSTM) deep learning algorithm is used to build and validate the web defacement detection method using the training data. The contributions of our paper include:

– Proposes to use features of plain text content extracted from web pages for differentiating defaced web pages from normal web pages;
– Builds and experiments the Bi-LSTM-based model for detecting web defacements using features of plain text content extracted from web pages. Extensive experiments on a large dataset of more than 96,000 web pages affirm that the proposed detection method produces much better performance than previous proposals.

This paper's remaining sections are structured as follows: section 2 reviews some closely related proposals. In section 3 depicts the proposed web defacement detection method and its processing stages, including preprocessing, training, and detection. In section 4 presents empirical outcomes and comments. Finally, section 5 gives the paper's conclusion and possible future tasks.

2. RELATED WORKS

As discussed in section 1, many proposals have been designed for detecting web defacements over the last decade. However, in the scope of this paper, we investigate some typical proposals of the group (3), which include proposals for website defacement detection using simple and complicated techniques. Web defacement detection proposals using classic techniques consist of DIFF comparison, checksum validation and document object model (DOM) tree dissection of web pages [11]. These approaches are comparably classic and extremely rapid; yet, they can simply handle web pages with fixed content or stable skeletons. Conversely, web defacement detection proposals using complicated techniques use more complex methods, including statistical methods, generic programming algorithms and machine learning techniques to build models for detecting web defacements. These approaches are commonly more sophisticated and resource-intensive. Nonetheless, the complex techniques-based proposals can be effectively used for monitoring and detecting web defacements on fixed web pages as well as dynamic pages. Particularly, related methods chosen for reviewing consist of proposals in [12]-[16].

Kim *et al.* [12], an approach based on statistics to detect website defacements is proposed. The approach uses the 2-gram method to construct a "profile" from a training dataset of normal web pages. The method is deployed in 2 stages, including the training stage and the detection stage. First, a "profile" is created by vectorizing the HTML code of each web page from the training dataset utilizing 2-gram sub-strings and their number of occurrences in the training phase. To represent each web page for defacement detection, 300 2-grams with the highest number of occurrences are chosen based on experiments. During the detection phase, the web page to be monitored is first retrieved, and its HTML code is transferred to a vector using a similar method as fulfilled for training web pages. The web page vector is then compared with the vector of the equivalent web page stored in the "profile" to determine the likeness score through the cosine distance. If this score falls below a pre-set detection threshold, an attack alarm is triggered. This threshold is first established and then periodically updated for each web page using an algorithm. However, the key drawbacks of the proposal are: i) the periodically updated thresholds may not be effective for web pages with frequently changed content, leading to a high number of false alarms; and ii) it demands significant computational resources to dynamically adjust thresholds for each monitored web page.

Bartoli *et al.* [13], Davanzo *et al.* [14], genetic programming algorithms are used to create the profile for detecting website defacement attacks. In the first step to gather web page data, they use 5 groups with 43 sensors to monitor and extract the monitored web page information. During the next step, the gathered web page information is transferred to the 1,466-element vector. The detection method is carried out in 2 stages: the training stage and the detection stage. During the training stage, data from normally operating web pages is first retrieved and then vectorized to compose the detection profile utilizing genetic programming algorithms. In the detection phase, the data of the monitored web page is first gathered, next vectorized, and then compared with the detection profile to identify any discrepancies. If a substantial discrepancy is found, an alarm is triggered. The primary drawback of this proposal is that it requires a significant number of computational resources to build the detection profile because computationally intensive genetic programming methods and large-sized web page vectors are used.

Hoang [11], a method for building web defacement detection models using traditional supervised machine learning techniques is introduced. In these models, the HTML code is vectorized for each web page using n-gram and term frequency (TF) methods. The approach utilizes an empirical dataset consisting of 300 defaced web pages and 100 normal web pages for model training and testing. Experiments with various models based on Naïve Bayes and J48 decision tree techniques, demonstrate that the proposed models achieve a high detection accuracy and low number of false alarms. Nonetheless, the major limitations of [11] are (1) the small size of the experimental dataset, which weakens the result reliability, and (2) the approach only analyzes the web page HTML code, leaving out other crucial embedded elements, such as images, JavaScript and CSS code.

To address the issues in [11], a mixed model for detecting website defacements is proposed. The approach combines the signature-based method and the machine learning-based method in a single detection model. In this model, the signature-based module first scans the HTML code of the monitored web page for pre-defined attacking signatures to improve performance against known defacement attacks. Then, the module based on machine learning classifies the web page HTML code using the classifier created during the training phase. In the end, the integrity validation of embedded files in the web page is done using the hashing technique. Experiments with the dataset of 1,200 defaced web pages and 1,200 normal web pages demonstrate that the mixed model delivers high detection results. However, while the model validates the embedded file integrity, the hashing method only works for static files. Building on their previous works [11], [15], a multi-layer model for detecting web defacement attacks is proposed [16]. In this model, a machine learning-based component is employed to detect defacements in the text content of web pages, such as HTML, CSS, and JavaScript code. In addition, the hashing method is applied for the integrity validation of embedded images. Experiments indicate that the mixed model effectively detects defacements in the web page text content. However, its ability to detect defacements in embedded images is limited, as it relies solely on hashing-based validation, despite the significance of embedded images in many web pages.

In short, the limitations of current proposals for detecting web defacements are as follows:
- Proposals based on classic methods, such as DIFF validation, checksum comparison and DOM-tree dissection can only handle web pages with fixed content or stable structures;
- Some proposals demand excessive computational resources since they use highly complex detection methods [13], [14];
- Some proposals produce a large number of false alarms and their detection results rely massively on the choice of pre-set detection thresholds [12];
- Some proposals can only handle web page text and/or HTML content. Other crucial web page components, such as CSS, JavaScript files and embedded images are unprocessed or processed using relatively basic methods, including hashing-based integrity validation [12]-[14].
- The datasets for experiments are pretty small, which decreases the result reliability [11]-[16].

3. PROPOSED MODEL FOR WEB DEFACEMENT DETECTION
3.1. Proposed model overview

The proposed model for detecting web defacements illustrated in Figure 2 is deployed in 2 stages, including (a) the training stage and (b) the detection stage. During the training phase as described in Figure 2(a), the training dataset is first collected by extracting text content in HTML files of defaced and normal web pages. Next, the text content of each page is separated into words using the tokenizer technique. Then, the dataset is trained using the Bi-LSTM deep learning algorithm [17] to produce a classifier.

In the detection phase as presented in Figure 2(b), the plain text content of the web page to be monitored is first retrieved. Next, it goes through a data preprocessing process similar to that of the training phase. Then, the web page processed text is classified utilizing the classifier constructed from the training phase to determine whether the status of the monitored web page is normal or defaced.

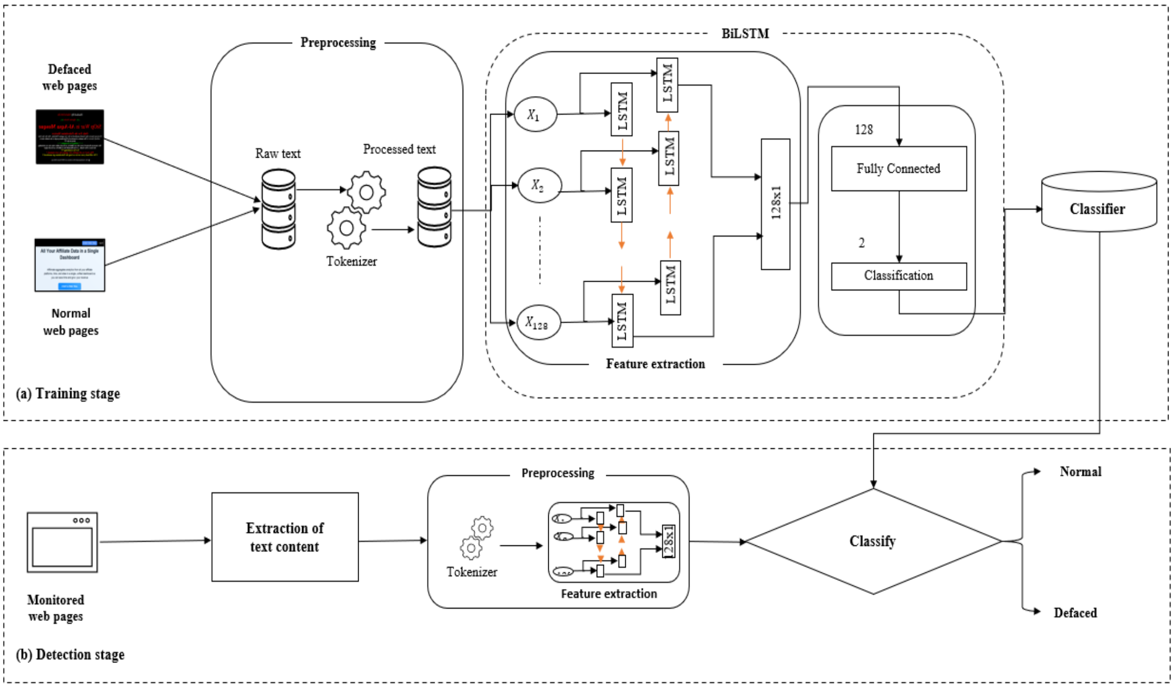

Figure 2. The proposed model for detecting web defacements: (a) training stage and (b) detection stage

3.2. Data preprocessing and model training

First, the plain text content extracted from each web page is preprocessed to retrieve features to form a vector representing that web page. Next, the preprocessed data is trained to create the model for detecting web defacements. The proposed model's data processing is done according to the 6 following steps:
- Step 1: a self-developed tool in Python is used to extract the plain text content from the combination of defaced web pages and normal web pages to create the training data. The data only consists of the plain text content of web pages, excluding HTML code and other forms of embedded code. Figure 3 explains

why the plain text content of web pages is selected as the feature to build the detection model. That is because the set of words that appears most commonly in plain text content extracted from defaced web pages as shown in Figure 3(a) is completely different from the set of words that occurs most frequently in normal web pages as shown in Figure 3(b).

– Step 2: from the obtained text data, the Tokenizer technique [18] is used to separate words in the text data and each word is transferred into a positive integer. Next, the first 128 consecutive words are chosen as the input to the Bi-LSTM algorithm. The number of 128 successive words for the input of the Bi-LSTM algorithm is chosen for the following reasons: i) on defaced web pages, the page content is often replaced by images and the text content of web pages is usually short. According to our survey, the number of web pages with text word count less than or equal to 128 is the dominant proportion for both defaced and normal web pages; and ii) the larger the number of words used the higher the requirement for computational resources, so it is necessary to choose an appropriate word space. Therefore, choosing a word space of 128 words as the input for the Bi-LSTM algorithm is suitable for the problem requirements.

(a) (b)

Figure 3. The top most commonly appearing words in text content extracted transformer from (a) defaced web pages and (b) normal web pages

The next steps are in the model training process using the Bi-LSTM algorithm as shown in Figure 4.

– Step 3: the embedding layer is used to help the model understand the semantic relationship of words through the model’s input vector. The result is a 128×128 vector that represents the characteristics of words and the relationships among words in the dataset, helping to increase the model’s ability to understand the web page’s text content.
– Step 4: the SpatialDropout1D layer is used to ensure that the model is not too dependent on some specific features and to help the model have better generalization ability on the data.
– Step 5: the Bi-LSTM layer is used to capture the association features among words to produce a vector of 128 features. Next, these 128 features will be the remaining features of the input for the fully connected layer.
– Step 6: the final fully connected layer converts 128 features into the model’s classification values using the SoftMax activation function to calculate the detection probability of defaced or normal web pages.

The Bi-LSTM deep learning algorithm was chosen to build the model for detecting web defacement attacks based on text content for 2 reasons: i) Bi-LSTM is a widely used deep learning algorithm in natural language processing on various problems, such as text classification and recommendation systems [19]-[22]. The Bi-LSTM model is structured with LSTM units that function in both ways, allowing it to integrate information from the past context and the future context. This enables Bi-LSTM to learn long-term dependencies without repeating information. Consequently, Bi-LSTM has proven highly effective for sequential modelling tasks and is extensively utilized in text classification. Unlike traditional LSTM networks, Bi-LSTM networks contain two parallel layers that propagate in forward and backward directions, capturing dependencies from both contexts [23]; ii) since the text content of defaced web pages usually has a relatively small number of words, the Bi-LSTM method with the nature of synthesizing information from two dimensions of text content is suitable for solving the problem in this paper.

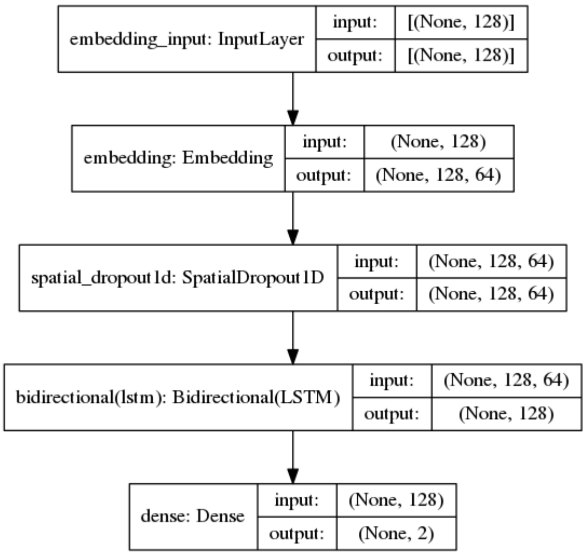

Figure 4. The structure of used Bi-LSTM algorithm

3.3. Performance measurements

The detection performance of the proposed model is measured using 6 measurements, including PPV, TPR, FPR, FNR, F1, and ACC. PPV is the precision and TPR is recall. FPR is the false positive rate and FNR is the false negative rate. F1 is the balance between PPV and TPR, and ACC is the overall accuracy. The following formulas are used to compute these measurements [24], [25]:

$$PPV = \frac{TP}{TP+FP} \tag{1}$$

$$PPV = \frac{TP}{TP+FP} \tag{2}$$

$$FPR = \frac{FP}{FP+TN} \tag{3}$$

$$FNR = \frac{FN}{FN+TP} \tag{4}$$

$$F1 = \frac{2TP}{2TP+FP+FN} \tag{5}$$

$$F1 = \frac{2TP}{2TP+FP+FN} \tag{6}$$

where, TP, TN, FP, and FN are parameters of the confusion matrix shown in Table 1.

Table 1. The confusion Matrix and its TP, TN, FP, and FN parameters

		Actual class	
		Defaced	Normal
Predicted class	Defaced	TP (true positives)	FP (false positives)
	Normal	FN (false negatives)	TN (true negatives)

4. RESULTS AND COMMENTS

4.1. Data collection and dataset building

The proposed detection model's experimental dataset is a combination of a set of defaced web pages and another set of normal web pages. Defaced web pages are downloaded from the Zone-h.org website [3]. Normal web pages are selected and downloaded from the top 1 million websites ranked by Alexa [26]. The process of web page collection is as follows: first, a tool is developed using JavaScript and NodeJS, then the tool is used to retrieve the HTML source code of web pages from their URLs; finally, the HTML source code of each web page is stored to an HTML file.

The collected dataset consists of (i) 57,134 HTML files of normal web pages labelled ‘0’ (normal) and (ii) 39,100 HTML files of defaced web pages labelled ‘1’ (defaced). The full dataset is divided randomly into 3 subsets of training subset, validation subset and testing subset as follows:
- The training subset that accounts for 60% of the full dataset is used as model input and model parameter tuning;
- The validation subset that accounts for 20% of the full dataset is used to check model accuracy during training to adjust parameters to avoid overfitting during training;
- The testing subset that accounts for 20% of the full dataset is used to evaluate the model after it has been trained.

4.2. Experimental results

The dataset described in section 4.1 is utilized to train, validate, and test the proposed defacement detection model. Additionally, we conduct experiments using detection models from [11] (Naïve Bayes-based and decision tree-based models) and [15] (random forest-based model) with the same dataset for a performance comparison against the proposed Bi-LSTM-based model. Table 2 presents the confusion matrix of the Bi-LSTM-based defacement detection model, while Table 3 highlights the detection results of the Bi-LSTM-based model alongside existing Naïve Bayes-based, decision tree-based [11], and random forest-based [15] models.

Table 2. The confusion matrix of the proposed web defacement detection model

Bi-LSTM-based web defacement model		Actual class	
		Defaced	Normal
Predicted transformerclass	Defaced	7,347	233
	Normal	433	11,234

Table 3. Detection results of the proposed model and existing models

Detection models	Features	ACC (%)	PPV (%)	TPR (%)	F1 (%)	FPR (%)	FNR (%)
Naïve Bayes-based model [11]	Text	82.54	78.12	79.26	78.69	15.21	20.74
Decision tree-based model [11]	Text	87.33	84.4	84.4	84.4	10.67	15.6
Random forest-based model [15]	Text	93.88	93.81	90.76	92.26	4.03	9.24
Proposed Bi-LSTM-based model	Text	96.54	96.93	94.43	95.66	2.03	5.57

4.3. Comments

There are some comments drawn from empirical results given in Table 3, as follows:
- The proposed Bi-LSTM-based detection model using plain text features produces significantly better detection measurements than that of the detection models proposed by [11] and [15]. Particularly, the ACC and F1-score of the proposed Bi-LSTM-based model, random forest-based model [15], decision tree-based model [11], Naïve Bayes-based model [11] are 96.54% and 95.66%, 93.88% and 92.26%, 87.33% and 84.4%, 82.54% and 78.69%, respectively.
- In addition, the FPR and FNR of the proposed Bi-LSTM-based model are also much lower than that of [11], [15]. Specifically, FPR and FNR of the proposed Bi-LSTM-based method, random forest-based method [15], and decision tree-based method [11] are 2.03% and 5.57%, 4.03% and 9.24%, and 10.67% and 15.6%, respectively.

The proposed Bi-LSTM-based model produces better detection results than previous proposals for the following two reasons: i) the selected features of plain text content extracted from web pages are suitable for detecting defacement attacks because hacking groups usually leave their the text signatures on defaced web pages after the attacks and ii) the chosen Bi-LSTM deep learning algorithm is suitable for processing text content data since Bi-LSTM algorithm is widely used in natural language processing. Furthermore, the Bi-LSTM algorithm includes LSTM units that work in both ways, enabling it to integrate information from the past context and the future context, which is highly relevant to the text content data of the problem in this paper.

5. CONCLUSION

This paper proposed a novel model for detecting web defacements. The model is based on the Bi-LSTM algorithm, utilizing plain text features retrieved from web pages. The proposed Bi-LSTM-based model produces a high detection rate and decreases the number of false alarms thanks to the Bi-LSTM’s capability to effectively handle text content extracted from normal and defaced web pages. Empirical results on over 96,000 web page dataset (including over 57,000 normal web pages and more than 39,000 defaced web pages) affirm that our detection model generates much better results compared to existing models on most performance measurements.

For future tasks, we will try to improve the proposed model or incorporate other features in the analysis and detection of web defacements to, i) enhance the detection rate and reduce the number of false alarms, especially the false negatives and ii) lower down the requirement of computing resources for the training stage and particularly the detection stage to improve practical applicability of the proposed method for detecting web defacements.

ACKNOWLEDGEMENTS

The authors deeply thank the Information Security Faculty, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam for the great assistance in fulfilling this project.

REFERENCES

[1] Imperva, "Website defacement attack," *Imperva*, 2021. https://www.imperva.com/learn/application-security/website-defacement-attack/.
[2] Trend Micro, "The motivations and methods of web defacement." https://www.trendmicro.com/en_us/research/18/a/hacktivism-web-defacement.html.
[3] "Zone-H.org." http://zone-h.org/archive.
[4] Acunetix, "Acunetix vulnerability scanner." https://www.acunetix.com/vulnerability-scanner/.
[5] Trustwave, "App Scanner." https://www.trustwave.com/Products/Application-Security/App-Scanner-Family/App-Scanner-Enterprise/.
[6] Misterscanner, "Vulnerability scanner." https://misterscanner.com.
[7] Vietnam Cyberspace Security Technology, "VNCS web monitoring service." https://vncs.vn/en/tin-tuc/detail-vncs-launches-the-first-online-website-monitoring-service-in-vietnam-281.
[8] L. Nagios Enterprises, "Web application monitoring." https://www.nagios.com/solutions/web-application-monitoring/.
[9] Site24x7, "Website defacement monitoring." https://www.site24x7.com/monitor-webpage-defacement.html.
[10] WebOrion, "WebOrion Monitor." https://www.weborion.io/monitor/.
[11] X. D. Hoang, "A website defacement detection method based on machine learning techniques," in *ACM International Conference Proceeding Series*, 2018, pp. 443–448, doi: 10.1145/3287921.3287975.
[12] W. Kim, J. Lee, E. Park, and S. Kim, "Advanced mechanism for reducing false alarm rate in web page defacement detection," 2006.
[13] A. Bartoli, G. Davanzo, and E. Medvet, "A framework for large-scale detection of Web site defacements," *ACM Transactions on Internet Technology*, vol. 10, no. 3, pp. 1–37, Oct. 2010, doi: 10.1145/1852096.1852098.
[14] G. Davanzo, E. Medvet, and A. Bartoli, "Anomaly detection techniques for a web defacement monitoring service," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12521–12530, Sep. 2011, doi: 10.1016/j.eswa.2011.04.038.
[15] X. D. Hoang and N. T. Nguyen, "Detectingwebsite defacements based on machine learning techniques and attack signatures," *Computers*, vol. 8, no. 2, p. 35, May 2019, doi: 10.3390/computers8020035.
[16] X. D. Hoang and N. T. Nguyen, "A multi-layer model for website defacement detection," in *ACM International Conference Proceeding Series*, 2019, pp. 508–513, doi: 10.1145/3368926.3369730.
[17] J. Brownlee, "How to develop a bi-directional LSTM for sequence classification in Python with Keras," *Neural Networks*, 2017. https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/.
[18] "TensorFlow." https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer.
[19] O. Melamud, J. Goldberger, and I. Dagan, "context2vec: learning generic context embedding with Bi-Directional LSTM," in *CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning, Proceedings*, 2016, pp. 51–61, doi: 10.18653/v1/k16-1006.
[20] T. Ceraj, I. Kliman, and M. Kutnjak, "Redefining cancer treatment: comparison of Word2vec embeddings using deep BiLSTM classification model," in *Text Analysis and Retrieval 2019: Course Project Reports (TAR 2019), pages 10–13, University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, July 2019*, 2019, no. 10–13, [Online]. Available: https://www.fer.unizg.hr/_download/repository/TAR-2019-ProjectReports.pdf#page=16.
[21] L. Xiao, G. Wang, and Y. Zuo, "Research on patent text classification based on Word2Vec and LSTM," in *Proceedings - 2018 11th International Symposium on Computational Intelligence and Design, ISCID 2018*, Dec. 2018, vol. 1, pp. 71–74, doi: 10.1109/ISCID.2018.00023.
[22] Y. Luan and S. Lin, "Research on text classification based on CNN and LSTM," in *Proceedings of 2019 IEEE International Conference on Artificial Intelligence and Computer Applications, ICAICA 2019*, Mar. 2019, pp. 352–355, doi: 10.1109/ICAICA.2019.8873454.
[23] B. Jang, M. Kim, G. Harerimana, S. U. Kang, and J. W. Kim, "Bi-LSTM model to increase accuracy in text classification: Combining word2vec CNN and attention mechanism," *Applied Sciences (Switzerland)*, vol. 10, no. 17, p. 5841, Aug. 2020, doi: 10.3390/app10175841.
[24] N. K. Sangani and H. Zarger, "Machine learning in application security," in *Advances in Security in Computing and Communications*, InTech, 2017.
[25] D. Gibert, C. Mateu, and J. Planes, "The rise of machine learning for detection and classification of malware: research developments, trends and challenges," *Journal of Network and Computer Applications*, vol. 153, p. 102526, Mar. 2020, doi: 10.1016/j.jnca.2019.102526.
[26] Alexa, "Top 1 million domains." http://s3.amazonaws.com/alexa-static/top-1m.csv.zip.

BIOGRAPHIES OF AUTHORS

Xuan Dau Hoang is an associate professor at the Faculty of Information Security, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam. He received a Ph.D. degree in Computer Science from RMIT University, Melbourne, Australia in 2006. Assoc. Prof. Hoang is the Dean of the Faculty of Information Security and Head of the Cyber Security Lab, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam. His current research interests include attack and intrusion detection, malware detection, system and software security, web security, and machine learning-based applications for information security. He can be contacted at email: dauhx@ptit.edu.vn.

Trong Hung Nguyen received a bachelor's degree in information technology in 2013 at the Academy of People's Security. He then received a master's degree in information security at the Academy of Cryptographic Techniques in 2018. He is currently a lecturer at the Faculty of Information Technology and Security, Academy of People's Security, Hanoi, Vietnam. Trong Hung is also a Ph.D. student at the Graduate University of Science and Technology, Hanoi, Vietnam. His research interests include attack and intrusion detection, malware detection and web security. He can be contacted at email: tronghungt31@gmail.com.

Hoang Duy Pham is a senior lecturer at the Faculty of Information Security, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam. He received a Ph.D. degree in Artificial Intelligence from the University of Queensland, Brisban, Australia in 2010. His current research interests include anomaly detection, malware detection, system and software security, web security, and machine learning-based applications for information security. He can be contacted at email: duyph@ptit.edu.vn.