JST: Smart Systems and Devices
Volume 34, Issue 2, May 2024, 010-017

VNeSafe: Machine Learning-Assisted System for Detecting Malicious URLs and Spam Calls

Van Tong[1], Dong Le Van [1], Quynh Anh Vu [2], Tuan Anh Ngo[3], Tri Duc Phung[4], Van Trong Nguyen[5], Duc Tran[1*]

[1]Hanoi University of Science and Technology, Ha Noi, Vietnam
[2] Faculty of Economical Mathematics, National Economics University, Ha Noi, Vietnam
[3]Smart Cyber Security Joint Stock Company, Ha Noi, Vietnam
[4]VMO Holdings Technology Joint Stock Company, Ha Noi, Vietnam
[5]Ministry of Public Security, Vietnam
[*]Corresponding author email: duc.tranquang@hust.edu.vn

Abstract

Spam calls and malicious Uniform Resource Locators (URLs) have become major concerns for Internet users. Phishing, spam, and drive-by-download attacks can be initiated by malicious URLs, while normal users may experience irritation from spam calls. To tackle the aforementioned issues, we provide VNeSafe, a machine learning-assisted system, in this paper. By leveraging user feedback, VNeSafe may identify a phone number that is spam. Particularly, it keeps track of how many times a phone subscriber has been reported as spam. When such a number is over a predetermined threshold, VNeSafe automatically adds the phone number to a blacklist and blocks it. Furthermore, VNeSafe uses a natural language processing technique named TF-IDF in order to extract good features from a URL. The Random Forest algorithm then makes use of these features to determine whether the URL is malicious or not. Our empirical research has demonstrated that Random Forest can offer a real-time detection with an F1-score of 0.9298. This algorithm is ready to be deployed in VNeSafe and used on a general mobile device.

Keywords: VNeSafe, malicious URL, spam call, Random Forest.

1. Introduction

Mobile devices have many of the capabilities of a traditional PC, as well as a wide range of connectivity options such as 4G, IEEE 802.11, Bluetooth, and NFC. The past decade has witnessed a widespread adoption of mobile devices. According to the Ministry of Information and Communications (MIC), by the end of 2021, Vietnam had 91.3 million subscribers, accounting for 73.5% of the country's adults [1].

Mobile users cannot be immune to cyberattacks. The most common types of attacks that a user may encounter are malicious Uniform Resource Locators (URLs) and spam calls. It has been observed that 39% of URLs are malicious [2]. Malicious URLs can be used to instantiate drive-by-download, phishing, and spam. A common practice is to detect malicious URLs by using a blacklist. Although the blacklisting method has a low false positive rate (FPR), it suffers when dealing with newly generated URLs. Besides, adversaries can utilize URL shortening services or fast flux to bypass the blacklist.

Machine Learning (ML) has emerged as a transformative force capable of solving a wide range of real-world problems. Machine learning enables computers to gain insights into a URL in order to extract relevant features, which are then used to determine whether the URL is malicious or not. The features can be either lexical or host-based [3]. The lexical features are obtained with the aid of Natural Language Processing (NLP) techniques such as Term Frequency - Inverse Document Frequency (TF-IDF) and Bag of Words (BoW) and Word2Vec.

Companies, on the other hand, are increasingly using phone calls to interact with their customers. Mobile marketing has evolved into an effective business tool. However, misuse of this medium can result in the proliferation of spam calls, potentially causing inconvenience to legitimate users. The Vietnamese government issued the 91/2020/ND-CP decree [4], which prohibits the transmission of advertising calls to users who have not given prior consent. The MIC organized, developed, and published a list of phone numbers that distribute spam on its official portal. As of April 1st, 2023, Vietnam telecom providers were required to deactivate phone numbers with incorrect or missing owner identity information. Despite the blocking of two million mobile numbers, many people reported that the amount of spam calls they received had not decreased [5].

ISSN 2734-9373
https://doi.org/10.51316/jst.174.ssad.2024.34.2.2
Received: Dec 13, 2023; revised: Mar 11, 2024;
accepted: Mar 14, 2024

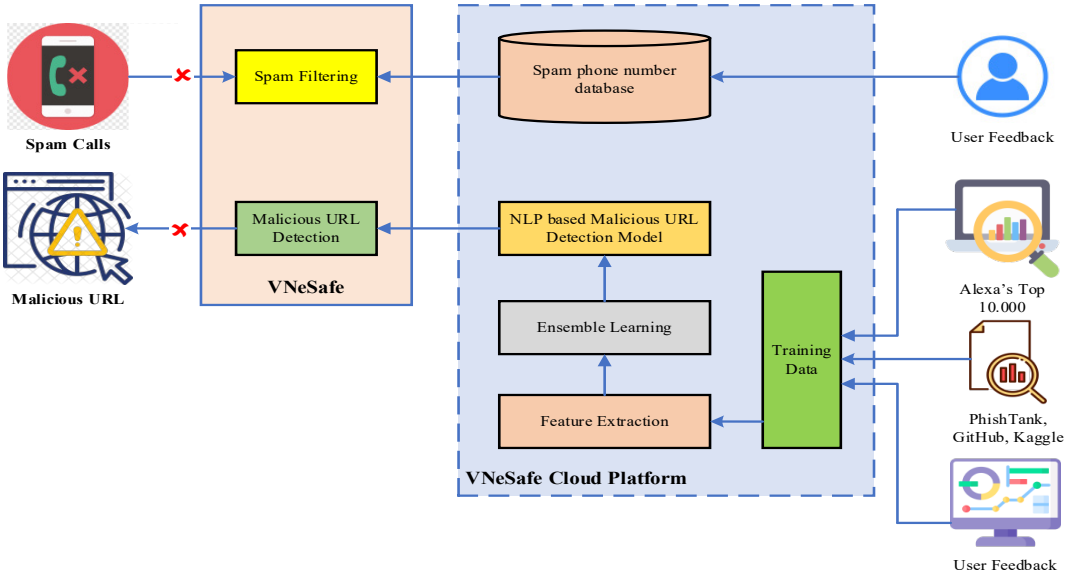


Fig. 1. General architecture of VNeSafe.

In this paper, we present a system, called VNeSafe, that is able to detect malicious URL and spam calls. Particularly, VNeSafe processes the incoming calls and informs the mobile devices about the span nature of the calls based on past user feedback. The system also has the capability of leveraging machine learning techniques in detecting malicious URLs with an accuracy of 93%. VNeSafe is lightweight as it takes 0.05 milliseconds to process a single URL. Hence, it is amenable to immediate applications.

The remainder of this paper is organized as follows: Section 2 introduces the general architecture of VNeSafe. Section 3 details the application of machine learning for detecting malicious URLs. The experiment results are discussed in Section 4. Section 5 is dedicated to conclusions and future work.

2. VNeSafe

The general architecture of the VNeSafe system is illustrated in Fig. 1. The system includes (1) an application that is installed on mobile devices, and (2) a cloud-based service that manages lists of machine learning models, malicious URLs, and spam phone numbers.

VNeSafe is capable of blocking spam calls. Spam calls, in particular, are detected using a set of valid and invalid signatures. These signatures would inform mobile devices about which calls should be blocked. This is a direct method of quarantining calls in which users specify a set of phone numbers from which they would like to see all calls encoded in blacklists blocked and another set of phone numbers from which they are always ready to receive calls encoded in whitelists. Whitelists are only stored on mobile devices to protect user privacy, while blacklists are stored in the VNeSafe cloud system.

Whitelists can be customized, which means that each user can specify his or her own valid phone numbers. Blacklists can be created using two different sources of data. The first can be obtained through reliable database on the Internet. For example, fraudulent international phone numbers that users should be wary of include +224, +231, +232, +247, +252, +375, +381, etc. The second source is generated directly by user feedback. When a user receives a spam call, he sends a spam feedback message to the VNeSafe cloud system, indicating that the current call was a spam call. It should be noted that the definition of a spam call is subjective to the user. VNeSafe counts the number of times a phone subscriber has been reported as spam. When this number reaches a certain threshold, the phone number associated with it is added to the blacklist.

Users can use VNeSafe to check if a URL is malicious. When a user receives an unusual URL, he or she can submit it to the VNeSafe system. The popularity of URL shortening services, which take a long URL as input and produce a short URL as output, makes detecting malicious URLs difficult. VNeSafe overcomes this difficulty by viewing URLs in a sandbox to acquire the true URLs. The URLs are then fed into a machine learning model on the VNeSafe cloud platform, which determines whether they are harmful or not. The classification results are returned to alert the users. The following section goes into greater detail about the machine learning model.

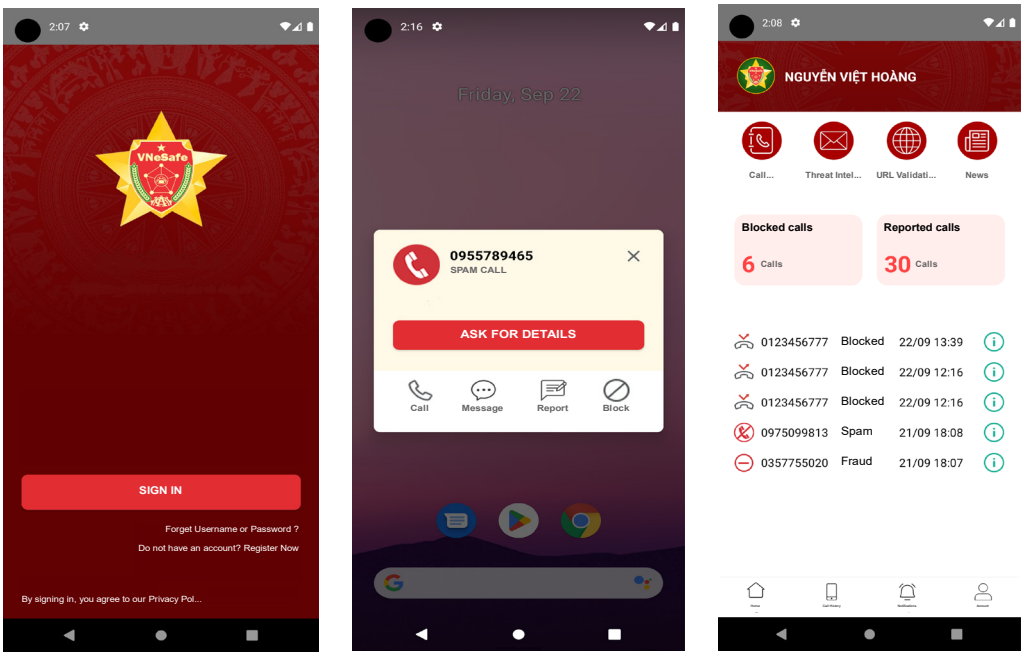


Fig. 2. Examples of the functionalities of VNeSafe

We emphasize that in this case, the use of machine learning has no effect on the overall performance of the device on which the VNeSafe application is installed. Machine learning models are trained on a dataset that contains both legitimate and malicious URLs. The legitimate URLs come from Cloudflare's Top 10,000 [6], whereas the malicious URLs come from PhishTank [7] and Github [16]. The model is trained on the VNeSafe cloud.

Other general features of VNeSafe include user verification, analytics dashboard, and push notifications. User verification can be accomplished through the use of a username/password, biometric data, and a one-time password. Analytics dashboard shows the number of calls received by the user as well as the number of calls blocked by his device. Push notifications are small pop-up messages that appear when an incoming call is suspected of being spam. Examples of these features are depicted in Fig. 2.

3. NLP-Based Malicious URL Detection

This section is dedicated to an in-depth analysis of the proposed malicious URL detection mechanism using NLP. The proposed approach contains two main modules: feature extraction and classification. The former considers a URL as input and extracts statistical and semantic features for the further module, while the latter analyses these features using ensemble algorithms such as Random Forest [8], Adaboost [9] and XGBoost [10]. The objective is to classify the URL as normal or malicious. Nowadays, with the rapid development of GPU (Graphics Processing Unit) and TPU (Tensor Processing Unit), many Deep Learning architectures are designed for text processing such as Transformer, DistilBERT, SecureBERT and so on [11]. However, these architectures demand a large amount of processing time, so it is not appropriate for VNeSafe, which requires real-time responses. Although the proposed URL phishing detection mechanism considers a lightweight solution with NLP techniques and ML algorithms, it can achieve good performance and require an appropriate processing time. The detail description related to this approach is illustrated in Fig. 1.

3.1. Feature Extraction

There are three common NLP techniques: BoW, TF-IDF and Word2Vec.

TF-IDF [12] is a statistical algorithm that illustrates how important a term is in a document compared to other documents in a collection of documents. TF-IDF is a multiplication of two parameters: Term Frequency (TF) and Inverse Document Frequency (IDF). The former describes how a word appears in a document, while the latter depicts how uncommon a word is in the collection of documents. In malicious URLs, there are many uncommon terms. Using TF-IDF can learn the statistical features related to the URLs in order to categorize them effectively. In URL phishing detection, URLs are split into different short parts by delimiter "/". Each part is considered a word, a URL is considered as a document, and a URL dataset is considered a collection of documents. The collection in TF-IDF is illustrated in Fig. 3.

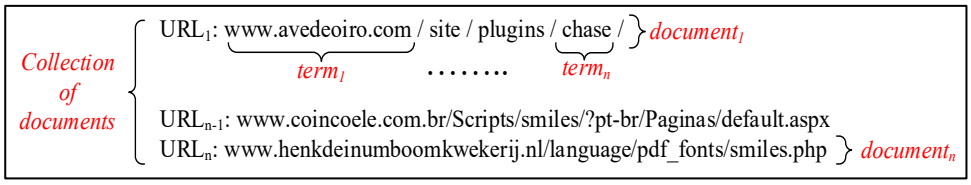


Fig. 3. The collection of documents in TF-IDF.

TF-IDF of term t in URL d is depicted as following:

$$TF\text{-}IDF(t,d) = TF(t,d) \times IDF(t) \quad (1)$$

where $TF(t,d)$ is the number of times term t appears in URL d, and IDF of term t is as follows:

$$IDF(t,d) = log[N/df_t] \quad (2)$$

where N is the total URLs in the dataset, and df_t is the number of URLs containing the term t.

BoW [13] is the simplest word-embedding technique which converts a sentence to a bag of word vectors. In the context of URL phishing detection, a URL is considered as a sentence which contains terms separated by "/". The URL is converted to a binary vector, and the length of the vector is the total number of terms.

Word2Vec [14] is a kind of word-embedding technique that represents vocabulary and enables the acquisition of word semantics, context, and its relationship. A word is considered as a term separated by "/" in URLs. TF-IDF and BoW learn the statistical features of a term in URL, while Word2Vec learns the context between consecutive terms. Word2Vec has two kinds of models: continuous bag-of-words (BOW) and continuous skip-grams. The first model predicts a term based on surrounding terms in URLs, while the second one considers a term and predicts the context of this term (other terms before and after this term). In URL phishing detection, the objective is to extract the semantics of consecutive terms, so we consider the skip-grams model.

3.2. Classification

After extracting features with NLP techniques, these features are analyzed in the classification module with ML algorithms. In VNeSafe, URL classification needs to achieve good performance and a low false prediction rate. If a normal URL is categorized as a malicious URL, VNeSafe will prevent this URL from doing illegal activities such as Distributed Denial-of-Service attack, data stealing, and so forth. However, it is a false alarm, influencing the perception of mobile users. Therefore, VNeSafe considers ensemble learning techniques in malicious URL classification detection in order to reduce false alarms. Ensemble learning method is an approach which generates many models and combines them to obtain improved results. There are two common approaches in ensemble learning: bagging and boosting [15]. The first approach trains the models on different subsets of the dataset and averages these models (e.g., using majority vote, etc.) to get the results, while the second one combines a set of weak learners into a strong one learner to minimize the error.

In the bagging approach, Random Forest (RF) algorithm [8] is taken into account. This algorithm is a classifier which creates many decision trees on different subsets of the dataset and takes average to enhance the classification results. The higher the number of trees, the higher the accuracy. Besides, a high number of trees can prevent overfitting problems. In the beginning, the URL dataset is divided into different subsets. Then, each subset is trained separately to create a corresponding decision tree. After obtaining results from these decision trees, the results are analyzed in the majority vote algorithm, which finds a majority result among results. Then, the majority result is assigned to the final result of RF. For example, if four out of five trees classify a URL as a normal URL, it will be assigned to normal. The details of the RF algorithm is shown in Fig. 4.

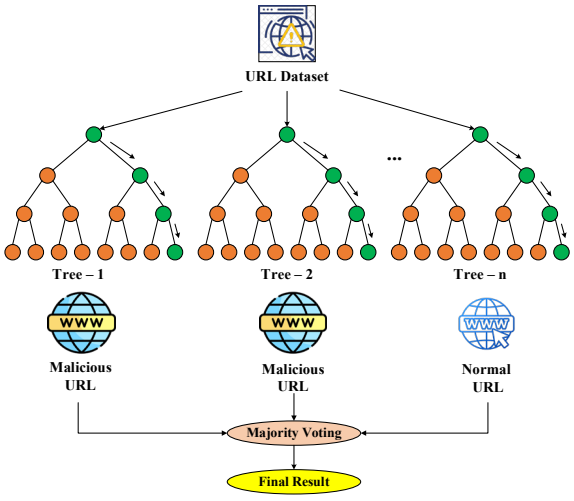


Fig. 4. Random Forest algorithm.

As for boosting approach, Adaboost [9] and XGBoost [10] are implemented for classifying URLs. Adaboost stands for adaptive boosting algorithm which was the first practical boosting algorithm.

Table 1. Performance analysis of different ML algorithms and NLP techniques.

Method	Precision	Recall	F1-score	Training Time (seconds)	Evaluation Time (milliseconds)
BOW + Native Bayes	0.9039	0.9001	0.8993	0.4012	0.0052
BOW + Decision Tree	0.9272	0.9269	0.9266	9.5361	0.1243
BOW + RF	0.9319	0.9318	0.9316	5.0120	0.0650
BOW + Logistic Regression	0.9223	0.9223	0.9222	3.5137	0.0460
BOW + AdaBoost	0.9185	0.9162	0.9163	260.59	0.0362
BOW + XGBoost	0.9213	0.9213	0.9213	175.77	0.0011
TF-IDF + Native Bayes	0.9039	0.9026	0.9022	0.0707	0.0009
TF-IDF + Decision Tree	0.9273	0.9272	0.9270	9.7156	0.1266
TF-IDF + RF	0.9298	0.9299	0.9298	3.9489	0.0515
TF-IDF + Logistic Regression	0.9233	0.9234	0.9233	3.3598	0.0437
TF-IDF + AdaBoost	0.9249	0.9249	0.9249	239.25	0.0333
TF-IDF + XGBoost	0.9237	0.9236	0.9236	169.65	0.0084
Word2Vec + Native Bayes	0.6089	0.6028	0.5962	0.4359	0.0057
Word2Vec + Decision Tree	0.5073	0.5073	0.5064	35.851	0.4672
Word2Vec + RF	0.4938	0.4938	0.4933	16.096	0.2098
Word2Vec + Logistic Regression	0.8321	0.8017	0.7978	2.6910	0.0350
Word2Vec + AdaBoost	0.8416	0.8248	0.8225	463.47	0.1366
Word2Vec + XGBoost	0.8302	0.8138	0.8114	323.66	0.0034

Unlike the bagging approaches which train decision trees in parallel, Adaboost trains decision trees sequentially. Concretely, the next decision tree is influenced by the previous tree. If the URLs are classified incorrectly in the previous decision tree, the weight of these URLs is increased in the next tree so that the next tree puts a special focus on them. Regarding the XGBoost, it is gradient boosting algorithm which uses accurate approximations to obtain the optimal tree model. Compared to Adaboost, XGBoost contains several benefits: parallelized tree building, tree pruning using Deep-First Search approach, cache awareness and regularization for preventing overfitting problem [10]. The details of Adaboost and XGBoost are depicted in Fig. 5.

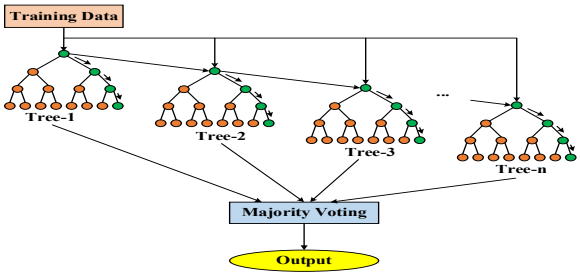


Fig. 5. Adaboost/XGBoost algorithm.

3. Experimental Results

This section is dedicated to explaining the experimental setup containing performance metrics and benchmarks.

3.1. Dataset

To evaluate the performance of the proposed approach and benchmarks, we use a synthesis dataset which is collected from Cloudflare [6], PhishTank [7] and Github [16]. The dataset contains 48,009 samples for normal URLs and 47,904 samples for malicious URLs. Each sample contains URLs and corresponding labels. The dataset is divided into two parts for training and testing with the ratio 80:20.

3.2. Experimental Setups

The malicious URL detection mechanism and benchmarks are evaluated using Precision, Recall, and F1-score. Precision is the percentage of retrieved URLs in a class, while recall is the percentage of retrieved URLs in relevant classes. F1-score combines precision and recall using a harmonic mean function. These parameters are calculated based on True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN), are described in following expressions:

$$Precision = TP/(TP + FP) \quad (3)$$

$$Recall = TP/(TP + FN) \quad (4)$$

$$F1\text{-}score = 2/(1/Precision + 1/Recall) \quad (5)$$

In this paper, we consider several ML algorithms containing Naive Bayes, Decision Tree, and Logistic Regression. Besides, the implementation of the proposed approach and benchmarks are written in Python and scikit-learn library. The experiments are implemented on a computer with Intel(R) Core i5-13400F @ 4.6 GHz, 64GB of RAM, NVIDIA GeForce RTX 3080 10GB, and Ubuntu 20.04. The source code is available upon request.

3.3. Performance Analysis

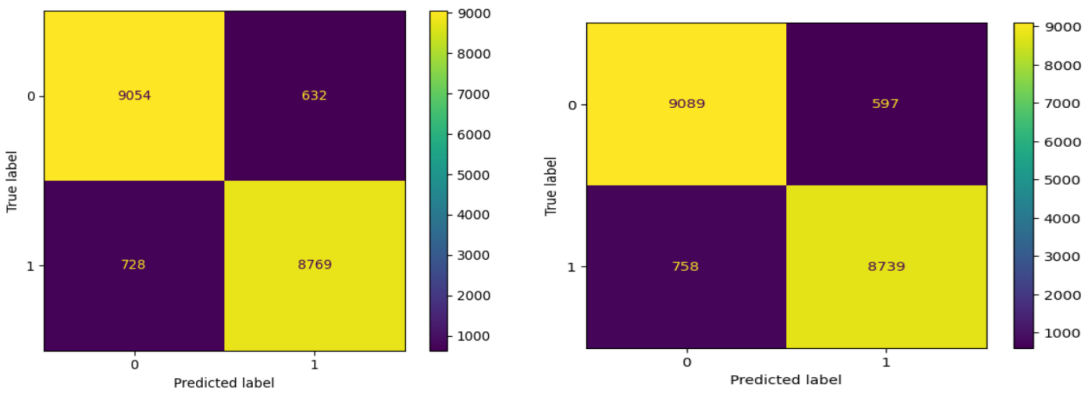
The performance of the malicious URL detection mechanism is evaluated with benchmarks with various NLP techniques and ML algorithms. The performance of these algorithms is depicted in Table 1. The performance with Word2Vec is the lowest in three considered NLP techniques, achieving F1-score from approximately 0.5 to over 0.82. Besides, TF-IDF and BoW can obtain better performance compared to Word2Vec, varying from 0.92 to 0.93. The reason is that TF-IDF and BoW use statistical features while Word2Vec uses semantic features. The malicious URLs contain many terms which do not have any meanings. Therefore, learning the semantic features is not effective. Moreover, in malicious URLs, there are many uncommon terms, so learning the statistical features can obtain better results in comparison with learning the semantic features. In TF-IDF and BoW, there is no significant difference between ensemble learning algorithms (RF, AdaBoost and XGBoost) and other ML algorithms (Logistic Regression, Naive Bayes, and Decision Tree). However, the F1-score of RF is slightly higher than the figure for other learning algorithms. BoW + RF can achieve 0.9316 of F1-score while TF-IDF + RF obtains 0.9298.

Although achieving better results than other ML algorithms, ensemble learning algorithms demand higher training and testing time. The training time of Adaboost, and XGBoost is the highest, ranging from 160 to 462 seconds. In contrast, the training time of RF is lower with a few seconds. A remarkable feature of Table 1 is that the performance of the testing time of considered algorithms is extremely small. BoW + RF only requires 0.065 milliseconds, while TF-IDF + RF demands 0.0515 milliseconds. The testing time is appropriate, so VNeSafe can provide the classification results in real-time.

To provide an in-depth analysis of the proposed mechanism for classifying malicious URLs, we compare the highest performance of the ensemble learning algorithm for each NLP technique. The details are shown in Table 2. Several normal URLs are incorrectly classified as malicious URLs, so the precision of BoW + RF and TF-IDF + RF varies around approximately 0.92. Therefore, the performance of BoW + RF is nearly equal to TF-IDF + RF, achieving approximately 0.93 of F1-score. The number of URLs classified correctly and incorrectly for these two approaches is given in Fig. 6.

Table 2. The highest performance of ensemble learning algorithm for each NLP technique.

URLs	BOW + RF			TF-IDF + RF			Word2Vec + AdaBoost		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Normal	0.9209	0.9420	**0.9313**	0.9196	0.9432	**0.9313**	0.7653	0.9361	0.8422
Unnormal	0.9405	0.9189	**0.9296**	0.9416	0.9173	**0.9293**	0.9180	0.7134	0.8029



(a) BOW + RF (b) TF-IDF + RF

Fig. 6. Confusion matrix of the proposed ensemble learning algorithm for URL phishing detection

Table 3. Comparison of VNeSafe and other call blocking applications.

Function	VNeSafe	Truecaller	CallApp	Whoscall
Caller ID	No	Yes	Yes	Yes
Spam call blocker	Yes	Yes	Yes	Yes
Offline database	Yes	No	No	Optional
Analytics dashboard	Yes	Yes	Yes	Yes
Automatic call recorder	No	No	Yes	No
URL scanner	Yes	No	No	Yes

3.4. Comparison with Other Call Blocking Apps

It should be noted that VNeSafe is not the only application that has the ability to block spam calls. Truecaller, CallApp, and Whoscall are among the other applications available. Table 3 compares the various call blocking apps. As can be seen, VNeSafe has a distinct advantage because it employs ensemble learning techniques to detect malicious URLs. In practice, this functionality is critical as it assists users in avoiding phishing and drive-by-download attacks.

4. Conclusions and Future work

We present VNeSafe, a machine learning-assisted system for detecting malicious URLs and blocking spam calls, in this paper. To detect spam phone numbers, VNeSafe relies on user feedback. It specifically counts the number of times a phone subscriber is reported as spam. When this number reaches a certain threshold, the phone number associated with it is added to the blacklist. Furthermore, VNeSafe used Random Forest to perform classification on URL-extracted features. The features are provided by the TF-IDF natural processing technique. Random Forest can provide real-time detection with an F1-score of 0.9298, according to experimental results. In the future, we plan to expand the functionalities of VNeSafe and release applications to the Google Play Store for users to use.

References

[1] Samaya. D., Vietnam Targets 85% Smartphone Usage by 2022. (accessed April 19, 2022), Online Avaiable: https://opengovasia.com/vietnam-targets-85-smartphone-usage-by-2022-end

[2] Ghalati, Nastaran Farhadi, Nahid Farhady Ghalaty, and José Barata, Towards the detection of malicious URL and domain names using machine learning, Technological Innovation for Life Improvement. Proceedings of 11th IFIP WG 5.5/SOCOLNET Advanced Doctoral Conference on Computing, Electrical and Industrial Systems, DoCEIS 2020, Costa de Caparica, Portugal, July 1-3, 2020. Springer International Publishing, 2020.

[3] Asiri, Sultan, Yang Xiao, Saleh Alzahrani, Shuhui Li, and Tieshan Li., A survey of intelligent detection designs of HTML URL phishing attacks, IEEE Access 11 (2023), pp. 6421-6443. https://doi.org/10.1109/ACCESS.2023.3237798

[4] Ministry of Information and Communications. Fighting Spam Messages, Spam emails and Spam Calls. Online Avaiable: https://thuvienphapluat.vn/van-ban/EN/Cong-nghe-thong-tin/Decree-91-2020-ND-CP-fighting-spam-messages-spam-emails-and-spam-calls/451726/tieng-anh.aspx (accessed December 13, 2021).

[5] Bao Ha Noi Moi. Continue to block spam calls and messages. Online avaibale: https://hanoimoi.vn/tiep-tuc-ngan-chan-cuoc-goi-tin-nhan-rac-450145.htmll (accessed July 28, 2022).

[6] Celso. M and Sabin. Z., Cloudflare Radar Domain Rankings, accessed September 30, 2022, Online Avaiable: https://blog.cloudflare.com/radar-domain-rankings/.

[7] Cisco Talos Intelligence Group. PhishTank. (accessed November 14, 2023). Online Avaiable: https://phishtank.org/

[8] Genuer, R., Poggi, JM. (2020). Random Forests. In: Random Forests with R. Use R!. Springer, Cham. https://doi.org/10.1007/978-3-030-56485-8_3

[9] Freund, Yoav, and Robert E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. European Conference on Computational Learning Theory. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995. https://doi.org/10.1007/3-540-59119-2_166

[10] Nalluri, Mounika, Mounika Pentela, and Nageswara Rao Eluri. A Scalable Tree Boosting System: XG Boost. Int. J. Res. Stud. Sci. Eng. Technol 7, no. 12 (2020): 36-51.

[11] Min, Bonan, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. ACM Computing Surveys 56, no. 2 (2023): 1-40. https://doi.org/10.1145/3605943

[12] Liu, Hao, Xi Chen, and Xiaoxiao Liu. A study of the application of weight distributing method combining sentiment dictionary and TF-IDF for text sentiment analysis. IEEE Access 10 (2022): 32280-32289. https://doi.org/10.1109/ACCESS.2022.3160172

[13] Lubis, Devi Hawana, Sawaluddin Sawaluddin, and Ade Candra, Machine learning model for language classification: bag-of-words and multilayer perceptron, Journal of Informatics and Telecommunication Engineering 7, no. 1 (2023): 356-365. https://doi.org/10.31289/jite.v7i1.10114

[14] Dharma, Eddy Muntina, F. Lumban Gaol, H. L. H. S. Warnars, and B. E. N. F. A. N. O. Soewito. The accuracy comparison among word2vec, glove, and fasttext towards convolution neural network (cnn) text classification. J Theor Appl Inf Technol 100, no. 2 (2022): 349-359.

[15] Mienye, Ibomoiye Domor, and Yanxia Sun. A survey of ensemble learning: Concepts, algorithms, applications, and prospects. IEEE Access 10 (2022): 99129-99149. https://doi.org/10.1109/ACCESS.2022.3207287

[16] Arunabell. G., Benign-phishing url classification using whois and lexical features. Online Avaiable: https://github.com/arunabellgutteramesh/benign-phishing-url-classification-using-whois-and-lexical-features (accessed August 30, 2019).