Learning URL Embedding for Malicious Website Detection

Xiaodan Yan, Yang Xu, *Member, IEEE*, Baojiang Cui, Shuhan Zhang, Taibiao Guo, and Chaoliang Li

***Abstract*—The emergence of artificial intelligence technology has promoted the development of the Internet of Things. However, this promising cyber technology can encounter serious security problems while accessing the internet. A malicious website can disguise itself as a normal website, and obtain users' private information. Thus, it is very important to detect malicious websites using tools such as machine learning (ML) algorithms, as these algorithms can help us to identify abnormal information hidden in the mass traffic more easily. Accordingly, many feature engineering tasks must be performed from memory, as a strong machine learning model is greatly improved with good features. In this article, we propose an unsupervised learning algorithm that learns URL embedding. We also explore some key parameters regarding a domain embedding model to obtain a good effect on domain features.**

***Index Terms*—Feature engineering, machine learning (ML), malicious websites, URL embedding (UE) model.**

I. Introduction

TRADITIONAL malicious website identification relies on manual rules, and the setting of these rules completely depends on human experience, such as in the setting of thresholds for the rules. Excessive subjective judgment can easily lead to the nonobjective judgment of malicious information in the entire system. Moreover, as the amount of data increases, the set threshold is not updated, therefore, more malicious information cannot be identified. To avoid the interference of human factors in the system, we adopt a method of machine learning (ML) and data mining (DM) for identifying malicious websites. For example, to address computer worms, we can use ML methods, including decision tree (DT), naive Bayes (NB) classifiers, and artificial neural networks to mine the pattern from communication between infected computers and the network and then analyze and detect the malicious behaviors. We use ML to conduct an in-depth analysis of complex and diverse data, thereby achieving substantial progress in ML in the context of intelligent data analysis. When using ML algorithm to identify malicious websites, it is necessary to use feature engineering to help the model find suitable features.

Kolter *et al.* [1] propose that ML and related techniques have been widely applied to realize traffic classification. Bengio *et al.* [2] proposed fighting the curse of dimensionality by learning a distributed representation for words, allowing each training sentence to inform the model regarding an exponential number of semantically neighboring sentences.

A new approach has been suggested for improving malware mutation detection, using deep learning [3], [4]. By converting malicious code into a gray image and using a convolutional neural network (CNN) to identify and classify the code, the features of a malicious image can be extracted. Moreover, a bat algorithm is suggested for solving the problem of data imbalances between different malware families, to increase the detection speed and improve the efficiency of the model.

The industrial information scenarios are facing serious creditability problems, as malicious users can repudiate the web contract and breach security [5]–[7]. The proliferation of Internet of Things (IoT) devices has led to an increase in malicious domain names. Recurrent neural networks (RNNs) are powerful tools that can provide sequence learning, and are robust to distortion and noise [8]. In addition, they can make full use of long-range contextual information. Therefore, it would appear that RNNs have the potential to address our challenges. Graves [9] provides a comprehensive framework for classifying and transcribing sequential data. However, traditional RNN have some disadvantages that it only consider the forward-to-backward construction and does not consider reverse direction.

The expansion of the mobile internet and the flourishing of big data technologies have made information extraction increasingly important, and higher requirements have been proposed for mining attack patterns from malicious behavior [10]–[12]. To extract effective information for detecting malicious behavior from a large amount of traffic, Duffield *et al.* [13] uses ML approaches to analyze association between traffic and packet, and then correlates alerts information from the packet level with feature vectors extracted from the same traffic source. The authors designed a framework for analyzing and discriminating

Manuscript received October 1, 2019; revised December 27, 2019 and February 14, 2020; accepted February 28, 2020. Date of publication March 3, 2020; date of current version June 22, 2020. This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 531118010454, in part by Guangdong Natural Science Foundation of China under Grant 2016A030313540, and in part by Guangzhou Science and Technology Program under Grant 201707010284. Paper no. TII-19-4484. *(Corresponding author: Yang Xu.)*

Xiaodan Yan, Baojiang Cui, and Shuhan Zhang are with the Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: xdyan@bupt.edu.cn; cuibj@bupt.edu.cn; zsh2016213631@bupt.edu.cn).

Yang Xu is with the Hunan University, Changsha 410082, China (e-mail: xuyang2007cs@163.com).

Taibiao Guo and Chaoliang Li are with the Hunan University of Technology and Business, Changsha 410205, China (e-mail: tigerguo1995@gmail.com; li_chaoliang@163.com).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2020.2977886

malicious traffic, along with steps for proof of concept. The accuracy of the candidate ML method, in terms of actual packet tracking, was evaluated and predicted. For ML methods, the forecast validity period is a problem, especially in resource-intensive web applications. The preliminary results showed that the performance loss was small over a period of a week or two.

Feature engineering is a process of transforming original data into features that can describe the data effectively, and the performance of a model built with these features can be optimized on unknown data. Feature engineering is the key to ML. The more features there are, the clearer the description of the dataset. However, this extra information will increase the time complexity of the experiment and the complexity of the final model, which will greatly increase the computation time and result in the "curse of dimensionality." To obtain good features, a significant amount of energy must be invested to explore the data. Moreover, it may make the model less universal. Our goal is to identify malicious domain names constructed as sequences of characters, which are essentially equivalent to time series. Therefore, a deep neural network based on time series may assist in effectively extracting the features of these malicious domain names, which can subsequently improve the performance of detection.

Feature selection, as a preprocessing step for ML, is effective for reducing dimensionality, removing irrelevant data, increasing learning accuracy, and improving the comprehensibility of results. However, the recent increases in the dimensionality of data pose severe challenges to many existing feature selection methods with respect to efficiency and effectiveness. Recently researchers introduce a novel concept called "predominant correlation," and propose a fast filter method for identifying relevant features as well as the redundancy among relevant features, without pairwise correlation analysis. The efficiency and effectiveness of their method is demonstrated through extensive comparisons with other methods, using real-world data with high dimensionality. Moreover, malicious websites may have some similar features, and we can mine remarkable correlation among these domains. There are few studies researching this aspect, and we propose a deep neural network for solving this problem.

In this article, we propose a new algorithm called URL embedding (UE) for exploring the correlations among different domains, and the coefficients among these URLs can be calculated using our system. The most important requirement for UE is to choose a distributed representation for the URLs. In this case, we consider that distributed representation is efficient because some feature engineering, such as one-hot representation, will lead to the curse of dimensionality, and the domain vectors are discrete and sparse. We focus on a distributed representation of domains, and obtain a low-dimensional vector with the help of a neural network.

For the UE model, we need to store the mapping between the URLs and their distributed representations. One evident disadvantage of the method is the complexity of the space, and many spaces are required to store the domain embedding model, as we need to store many dimensional vectors in the memory. We explore the appropriate dimensions for domains, and ultimately choose a dimension. In this article, we propose that malicious websites can be treated as words, and a distributed representation for malicious websites can be trained using domain name system (DNS) queries.

The novel unsupervised learning algorithm proposed in this study is more effective than existing methods for ensuring cyber safety and reliability. Accordingly, we theoretically analyze the performance of the UE and optimize the algorithm. The contribution of this research is threefold as follows.

1) We mainly focus on cyber security problems caused by the surge of malicious website. We put forward a mechanism based on unsupervised learning to overcome the feature subjective problems which significantly improves the performance of malicious information identification.
2) We theoretically analyze the performance of the UE algorithm. The optimization result shows that the presented algorithm can achieve superior performance compared to a feature engineering classifier.
3) We solve the problem of the interference of human factors generated by the feature engineering classifier, making cyber devices more trustworthy for users. Based on the above, we study how to improve the stability of the proposed method, and experimentally verify its effectiveness.

The rest of this article is organizes as follows. Section II presents some related issues. Section III provides the detailed information about distributed representation of URLs and proposes a new concept which is called as UE model. Section V analyzes experiment results of the proposed model and some useful conclusions can be obtained. Finally, Section VI concludes this article.

II. Related Work

Large amounts of security scenarios adopt ML to assist the identification of malicious websites on a large scale. MapReduce is a distributed computing framework for processing massive-scale data sets for ML.

Ma *et al.* [14] explores detecting malicious websites from the lexical and host-based features of their URLs. They develop a real-time system for gathering URL features, and pair it with a real-time feed of labeled URLs from a large webmail provider. From these features and labels, they are able to train an online classifier for detecting malicious websites with 99% of precision over a balanced dataset.

Yadav *et al.* [15] present a method to mine the inherent patterns behind the domain names generated by the botnets. They employ statistical methods to distinguish the malicious domain names. They reach satisfying performance for different scenarios. Plohmann *et al.* [16] analyze the domain names generated by domain generation algorithms (DGA) and realize the accurate identification of malicious domain names by pre-computing future DGA domain names, using a taxonomy elaborately designed for DGAs. Samuel *et al.* [17] developed a malicious domain name detection system named "FANCI." It functions by categorizing the domain names. They deploy their system in a large-scale university network and successfully discover the malicious domain names generated by DGAs, which makes full use of the computing ability of the sever cluster,

extracts features from the malware, and uses them to train the one-class support vector machine (SVM) offline.

Millions of new domains are registered every day, and many of them are malicious. It is challenging to keep track of malicious domains using only web content analysis, owing to the large number of domains [18], [19]. He *et al.* [20] demonstrated that it is possible to transform this intuitive observation into statistically informative features, using second-order Markov models. In their study, four transition matrices are built from known legitimate domain names, known malicious domain names, and English words in a dictionary, based on a uniform distribution. The probabilities from these Markov models, as well as other features extracted from DNS data, are used to build a random forest classifier. The experimental results demonstrate that their system can quickly catch malicious domains with a low false positive rate.

Buczak *et al.* [21] describes a focused literature survey that investigates the network analysis methods of ML and DL to support the intrusion detection. Based on the number of citations or the relevance of an emerging method, papers representing each method are identified, read, and summarized. Because data are so important in ML/DM approaches, some well-known cyber data sets used in ML/DM are described. The article addresses the complexity of ML/DM algorithms, presents a discussion of challenges in using ML/DM for cyber security, and provides some recommendations for when to use a given method [22].

Blum *et al.* [23] explore the possibility of utilizing a confidence-weighted classification combined with content-based phishing URL detection, to produce a dynamic and extensible system for the detection of present and emergent types of phishing domains. Their system is capable of detecting emerging threats as they appear, and subsequently can provide increased protection against zero-hour threats, unlike traditional blacklisting techniques (which function reactively).

To reduce the uncertainty in artificial neural network modeling and realize the optimization of an open-loop trajectory, a stochastic modeling programming method was proposed, based on deep Bayesian neural networks [24]. Using an accurate long-term estimation of the system, the research proposes an offline planning for sample data, in combination with the optimization ability of a multistep prediction training model. Ultimately, the scheme realizes faster convergence and improved generalization ability.

As the certificateless lightweight signature (CLS) scheme can no longer guarantee its security performance against the four types of signature forgery attacks, a robust CLS (RCLS) scheme has also been proposed [25]. The RCLS scheme only requires the use of public channels, and can withstand public-key replacement attacks and malicious third parties, ensuring its reliability. Moreover, according to the performance evaluation, the RCLS scheme is far more stable than the CLS scheme, and is more suitable for IoT application. In addition, because the CLS schemes for the media transfer protocol (MTP) and random oracle model (ROM) lack guarantees of data security, an RCL scheme without MTP and ROM is proposed to guarantee the reliable security of data in the cloud-assisted IoT. Finally, the experimental results also confirm that the RCLS scheme is safe and effective.

Through investigation, it was found that many researchers focus on finding good features [26]–[28]. When the right model is found, it is the features that determine the upper limit of the model. Hall *et al.* [29] deal with both continuous and discrete problems by adopting the correlation-based filtering algorithm. The algorithm often out-performs the well-known Relief F attribute estimator when used as a preprocessing step for NB classification, instance-based learning, DT, locally weighted regression, and model trees. It performs more feature selection than Relief F, reducing the data dimensionality by 50% in most cases. Moreover, the decision and model trees built from the preprocessed data are often significantly smaller.

Recently, there have also been research studies exploring the concept of using good eigenvectors to assist the model in making better decisions. Radford *et al.* [30] show that a deep convolutional adversarial pair learns a hierarchy of representations from object parts to scenes, in both the generator and discriminator. Additionally, they made full use of the learned features for novel tasks, demonstrating their applicability as general image representations.

Researchers present several extensions for improving both the quality of the vectors and the training speed. By subsampling the frequent words, they obtain a significant increase in speed, and also learn more regular word representations. An inherent limitation of word representations is their indifference to word order, and their inability to represent idiomatic phrases. They present a simple method for finding phrases in text, and show that learning good vector representations for millions of phrases is possible.

Traditional methods for identifying malicious websites rely on a manually summarized rule system, which requires a rule-maker with a very strong security background. Moreover, the rule system is not robust. The threshold set by the original rule-maker will become very fragile when the data scale is large. As many malicious websites can be hidden in a large amount of traffic, the system will be unable to recognize them.

III. Distributed Representation for URL Embedding

Field representation has long been accomplished via feature engineering, where researchers aim to design better features for specific tasks. The most widely used feature representations are thermal and distributed representations. However, one-hot representation does not take into account the relevance and order of words, and risk of the curse of dimensionality. The domain vectors cannot describe the similarity between words, which lead to poor generalization ability of the model. To generate a low-dimensional vector for the URL, we choose a neural network to train a domain embedding model. As a difference from a back propagation neural network, we do not choose a hidden layer in the domain embedding model, but rather use Huffman coding to encode the URL. As compared with a traditional neural network, the training time for the UE model is shorter.

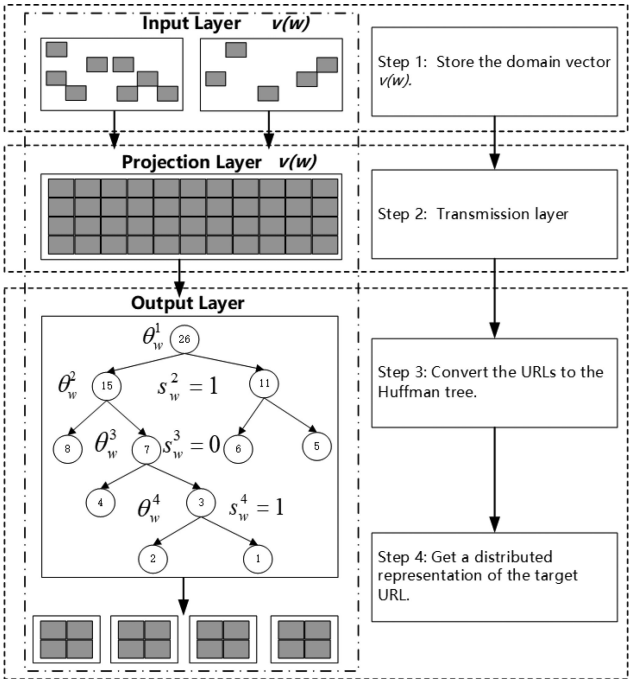

Fig. 1. Overall structure about UE model.

A. Structure of URL Embedding

There are three components in the UE model: the input layer, the projection layer, and the output layer. The input layer, which is the fundamental layer, only involves the domain vector $v(w)$. The core component is the output layer because the model encodes the domain using Huffman coding. Our goal is to get a distributed representation of the target URL. The projection layer is the transmission component connecting the input and the output. Fig. 1 reveals the overall structure of UE.

The input layer corresponds to the embedding information of URLs around the current URL, and the projection layer conducts sum pooling of the embedding information of the input layer, which is then input into the Huffman tree of the output layer. Then, the path from root to leaf node of Huffman tree was used for multiple dichotomies to finally obtain the current UE information of leaf node.

Huffman coding is a good encoding algorithm for compressing and shortening the length of the character encoding, with the help of character frequency. There are many URLs on the Internet, and we convert the URLs into the Huffman trees leaf nodes to draw support from the occurrence times of domains. We can build a URL dictionary "D," which contains a mapping between domains and domain vectors from the domain embedding model.

The Huffman tree will be built when the process of mapping is finished. A heap will be considered as a tool for storing leaf nodes and nonleaf nodes, with the rule that the order of weights is from largest to smallest. In Algorithm I, the structure of this process is detailed, as follows.

In Algorithm 1, each entry is a data structure storing the mapping between the domain and the frequency of the domain. We convert the results produced by the vocabulary factory into many entries, and push them into the heap. Each iteration, the program chooses the two nodes with the smallest weights, and combines them into a new node that will be pushed into the heap. The entire process continues until only one element in the heap, i.e., the root of the Huffman tree.

There are several paths from a root node to leaf nodes, and we need to obtain the Huffman codes of discrete domains. Algorithm 2 is a recursive method for generating the Huffman codes among all of the leaf nodes. Pos is an array which contains the Huffman code from root to leaf, in Algorithm 2, the Huffman codes will be generated in case of tree_root has no children.

The Algorithm 3 solves the problem that we need to obtain distributed representation of domains based on the Huffman code. P$(context(w)|w)$ is used to calculate the correlation

Algorithm 1: Build Huffman Tree.

Input: $domains$, $counts$, len.
Output: $tree\_root$
1) $heap =$ initialize_heap().
2) **for** $i = 1$ to len **do**
3) Heap.insert(entry($domains[i], counts[i]$)).
4) **end for**
5) **while**($heap$.size()>1)
6) $left\_child = heap$.pop().
7) $right\_child = heap$.pop().
8) $new\_node =$ entry($\wedge$, $left\_child.counts + right\_child.counts$).
9) $new\_node.right\_child = right\_child$.
10) $heap$.push(new_node).
11) **end while**
12) $tree\_root = heap$.pop().

Algorithm 2: Generate Huffman Code.

Input: $tree\_root$, pos, $level$.
Output: $tree\_root$
1) **if** ($tree\_root == null$)
2) return.
3) **end if**
4) **if** ($tree\_root.left\_child! = null$)
5) $pos[level] = 1$.
6) generateHuffman($tree\_root.left\_child, pos, level$+1).
7) **end if**
8) **if** ($tree\_root.right\_child! = null$)
9) $pos[level] = 0$.
10) generateHuffman($tree\_root.right\_child, pos, level$ +1).
11) **end if**
12) **if** ($tree\_root.left\_child == null$ && $tree\_root.right\_child == null$)
13) $seq =$ " ".
14) **for** $i = 1$ to $level$ **do**
15) $seq += pos[i-1]$.
16) **end for**
17) **end if**
18) $tree\_root.codes = seq$.

Algorithm 3: URL Embedding.

Input: $context(w)$.
Output: $v(w)$
1) $e = 0$.
2) **for** $u \in context(w)$ **do**
3) **for** $j = 1$ to l^u **do**
4) $s = \text{sigmoid}(v(w)^T \theta^u_{j-1})$.
5) $grad = \partial(1 - s^u_j - s)$.
6) $e = e + grad * \theta^u_{j-1}$.
7) $\theta^u_{j-1} = \theta^u_{j-1} + grad * v(w)$.
8) **end for**
9) **end for**
10) $v(w) = v(w) + e$.

scores among the target suspected URLs. Thus, we need to maximum the probability of $\text{P}(context(w)|w)$. In that regard, $context(w)$ represents the domains similar to the target domain w. In Algorithm 3, we conclude the update rules about the calculation of UE model.

To achieve UE algorithm, the current algorithm needs to be run for URL access sequence. This method mainly optimizes the output layer of softmax in the traditional neural network word language model. Because the URLs are sparse and the overall size is large, the computation from the hidden layer to the output layer is very heavy, and because we want to calculate the softmax probability of all the URLs, we should find the value with the highest probability. To avoid calculating softmax probabilities for all words, this article uses Huffman tree instead of the hidden layer to output softmax layer mapping. Since we changed all the previous probability calculations from the output softmax layer to a binary Huffman tree, our softmax probability calculations need only go along the tree structure. However, the left and right nodes of the binary tree can be treated as dichotomies, so the calculation of UE is equivalent to walking from the root to the corresponding leaf nodes. The entire path can be seen as multiple dichotomies, and finally the embedding information of URL can be obtained by maximizing the probability product of the path from root to leaf.

IV. Performance Evaluation

In this section, we evaluate the proposed UE model and traditional feature engineering. We use different feature construction methods to identify malicious websites, and collect passive DNS traffic data from real online network traffic. MapReduce and its variants have been highly successful for implementing large-scale data-intensive applications on commodity clusters. However, most of these systems are built around an acyclic data flow model, which is not suitable for other popular applications. The experiments demonstrated that the methodology is able to provide accurate and efficient identification.

A. Datasets

We experimented with open data, i.e., 360 NetLab and Alexa open dataset, to evaluate UE model. We choose Alex top 10 000 domains as our positive samples and some abnormal domains in 360 netlab as our negative samples. Feature engineering and UE can help us to convert the URLs to feature vectors, and we can test the performances of feature engineering and UE. The details are as follows.

TABLE I
Positive Samples and Negative Samples

Positive samples	Negative samples
Google.com	1yb3mkw1vipc2qt1mv4qr3xcqf.org
Sina.com	1f3yeryza1ulk1vuyrdw1nek6dd.com
qq.com	egkcoc1oay5hij4j78qgo8fbk.net
Youtube.com	1cj5mni164n5xqx2hvjiuyzpvf.com
Baidu.com	1qoqlc84ov1ax11dyg3h1y5y2xt.com
Facebook.com	1rlyqqjmg96163qi2bcn6hzkx.org
Sohu.com	1wakafb1qxf5jpl3mhp510bghi2.com
Yahoo.com	1nj3ubrxjm9p317bdrm3dcul8x.org
Sina.com	1f3yeryza1ulk1vuyrdw1nek6dd.com
Amazon.com	14t5kg6184p31fpjzi8yss8dfq.org
Taobao.com	9vw9k51jl2kgdk5y69k1bj6121.org

The identification of malicious websites here is mainly a dichotomy that requires positive and negative samples. With alexa, all the data in this website are normal URL information. We chose the 10 000 with the highest frequency as a positive example. Every day, 360 netlab will mine a large amount of malicious information on the Internet through their methods, taking it as a negative example. In this way, the trained classifier can better identify the malicious information.

B. Experiments and Results

In the process of selecting features for ML methods, the lexical features of URLs are chosen, as they are easy to analyze and can be widely applied to any types of malicious websites. The lexical features are listed as follows.

1) ***Length (l):*** Simple style is the characteristic of a normal domain name. Compared with a DGA URL, the length of normal URL is usually shorter to keep compact and comprehensible.
2) ***Vowels (v) and Percentage (v%):*** We take vowels into consideration which make URLs easier to understand.
3) ***Digits (d) and Percentage (d%):*** A significant amount of numbers in a URL often means that they have been generated automatically. URLs become less comprehensible when digits are involved, especially when they are presented without order.
4) ***Normal Suffix Test and Malicious Suffix Test:*** Whether it is normal or malicious, the URLs must contain suffixes, e.g., "com", "gov", "org", "edu", etc. Websites usually have URL suffixes that we are familiar with. Thus, we extract all the suffixes, including normal URLs and malicious URLs, to serve as features for training. Each suffixes is labeled by a number, malicious suffixes for "1" while normal suffixes for "0".

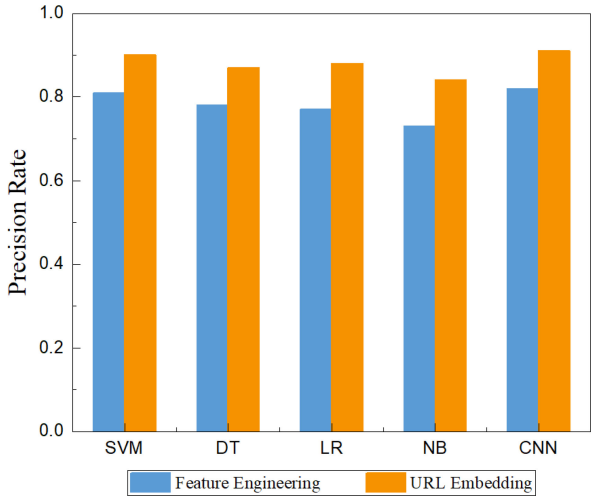

Fig. 2. Precision rate comparison.

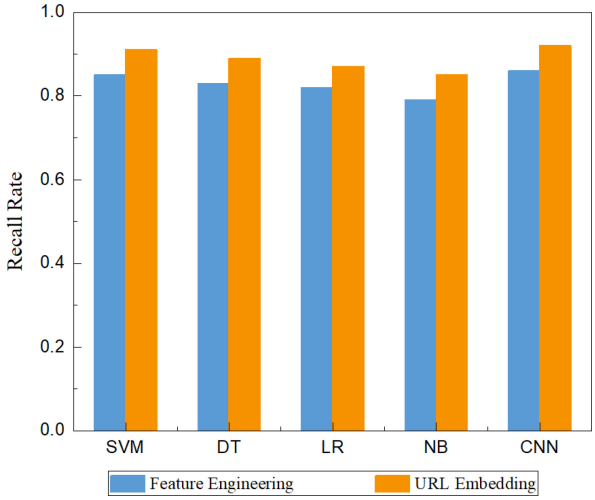

Fig. 3. Recall rate comparison.

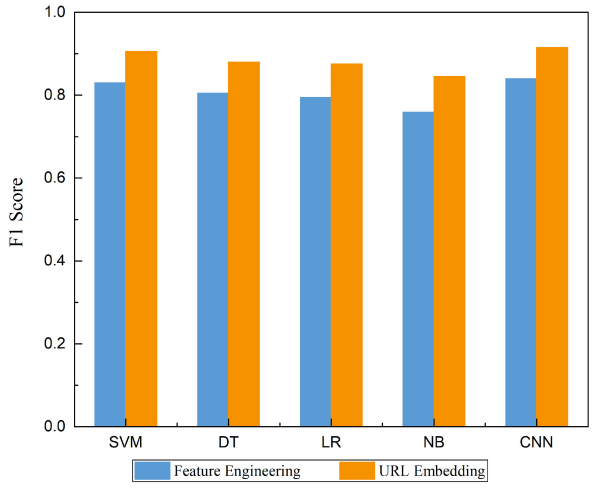

Fig. 4. F1 score comparison.

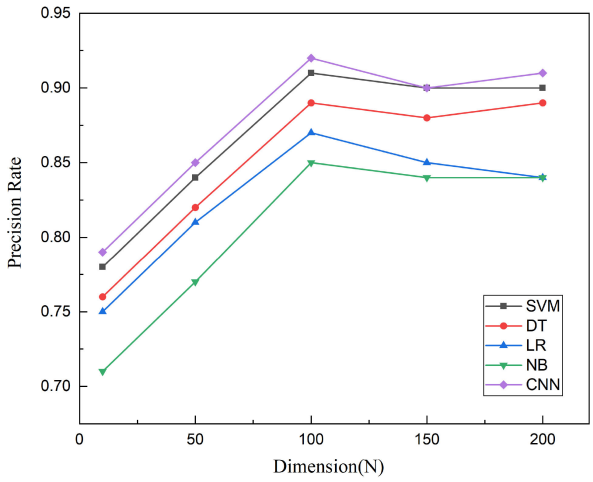

Fig. 5. Precision rate comparison with change of dimension N.

To better demonstrate the superiority of our model, we evaluate the SVM, DT, logistic regression (LR), NB, and CNN and compare them based on UE and feature engineering. We use precision rate, recall score and F1 score as the evaluation metrics. Identifying malicious messages in massive amounts of traffic information requires robust ML capabilities and a highly automated ML system. Feature extraction is performed by feature engineering and UE, and then the features are input into the classifier to evaluate the accuracy and recall rate of the overall prediction, respectively. Recall score represents the ratio of the number of real positive examples predicted by the algorithm to the number of all real positive examples. F1 score represents the harmonic average of precision rate and recall rate.

It can be seen from Fig. 2 that the precision rate of UE classifier is 9–10% higher than that of feature engineering classifier. It can be seen from Fig. 3 that the recall rate of UE classifier is 5–6% higher than that of feature engineering classifier. It can be seen from Fig. 4 that the F1 score of UE classifier is 7–8% higher than that of feature engineering classifier.

Considering the above criteria, UE classifier significantly improves the performance of malicious information identification compared with feature engineering classifier. The above experiments mainly evaluate the accuracy, recall rate, and F1 score.

For the UE model, we need to store the mapping between domain and distributed representations of domain. One evident disadvantage is the complexity of the space, a significant amount of space is required to store the domain embedding model, as we need to retain many N-dimension vectors in the memory. To explore appropriate N values, we conduct some experiments, and the result of experiments can be shown in Fig. 5–7.

To find the best dimension N, we consider the results of experiments performed with dimension vector of 10, 50, 100, 150, and 200. When $N = 10$, and using the SVM as an example, the precision of our model is 78%. With the N increasing to 100, the model precision increases to 91%. Other algorithms follow the same trend. We choose the $N = 100$ parameter as our final value, as the precision rate, recall rate, and F1 score were superior to those of other parameters. The detailed results

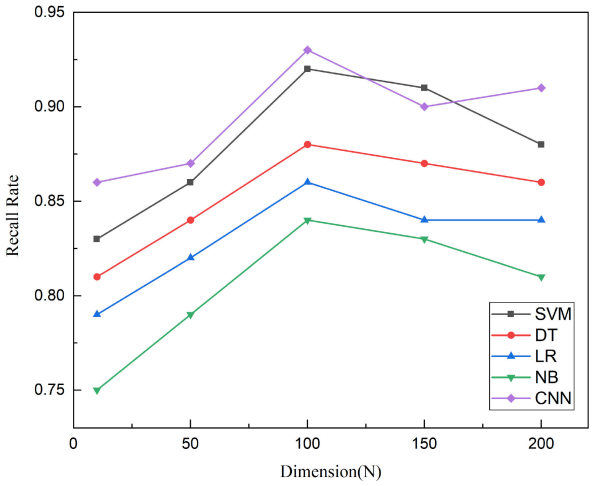

Fig. 6. Recall rate comparison with change of dimension N.

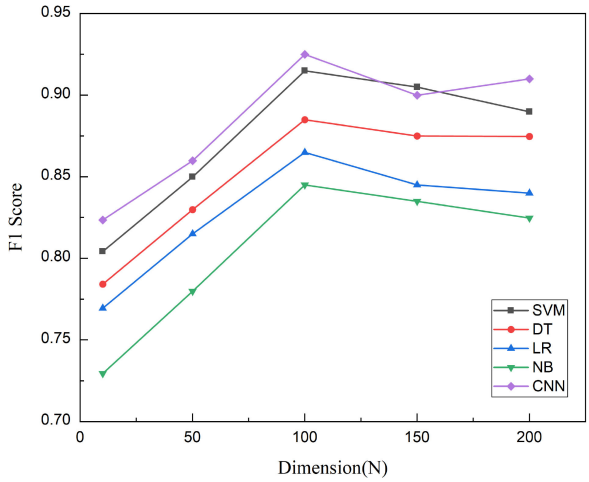

Fig. 7. F1 score comparison with change of dimension N.

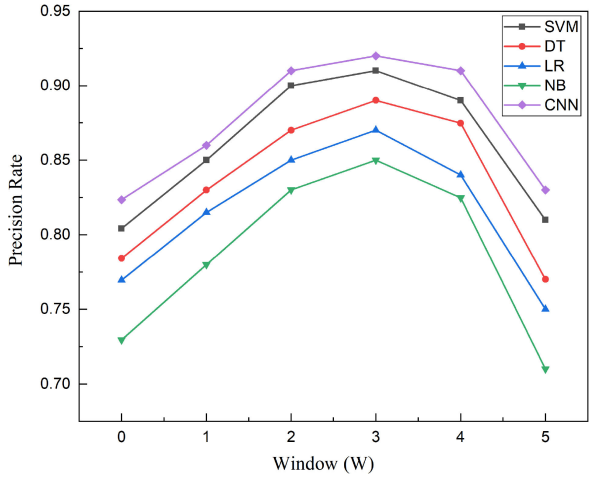

Fig. 8. Precision rate comparison with change of window W.

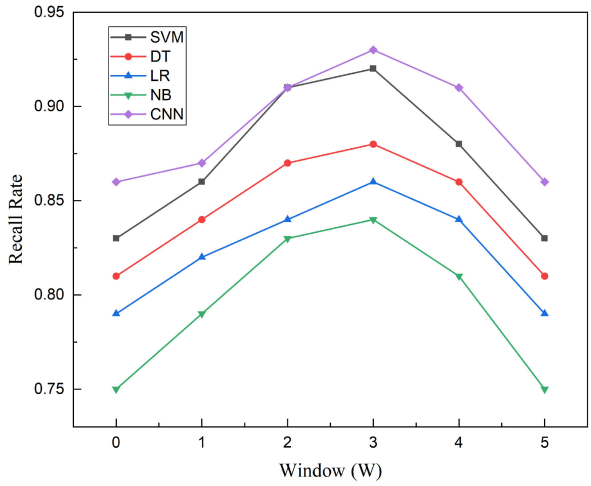

Fig. 9. Recall rate comparison with change of window W.

are illustrated in Fig. 5–7. We use the context information of the current URL to extract the embedding information of the current URL.

The context information of the URL is mainly the time series information of the website visited by the user who visits the current URL. As the context is the sequence information of the URL, to explore the most appropriate context, we need to explore the context information of the current URL, and to obtain the appropriate context window W. The window here represents the distance of the farthest URL to the left or right. In practice, if the W is 5, it is equivalent to modeling 5 URLs on the left and 5 URLs on the right of the current URL and completing sum pooling. We found through the experiment that the farthest choice is 5. After 5, basically the results are very similar.

This research introduces a novel unsupervised learning algorithm that trains the UE instead of the feature engineering jobs. We evaluate UE model in several ways. The outcomes of the evaluation indicate that the UE model can effectively reach the global optimization result. Meanwhile, we have successfully solved the problem of mining malicious URL from large quantities of data generated by hundreds of industrial devices. To find the appropriate context window, we consider the results of experiments performed with different windows representing the numbers of sequence information of the URL, and evaluate using values from 0 to 5. When $W = 0$, and using the SVM again as an example, the precision of our model is 80.42%. With the window increasing to 3, the model precision increases to 91% accordingly. Other algorithms follow the same trend. We choose the $W = 3$ as our final result, because the precision rate, recall rate, and F1 score were superior to those of other parameters. The detailed results are illustrated in Fig. 8–10.

In feature engineering, data are analyzed by people themselves to obtain corresponding features, and then the features are brought into the relevant models for classification or regression. The manual analysis of features has three disadvantages: 1) It costs a lot of manpower, requires a thorough understanding of the business, and has a high threshold for entry. 2) Many characteristics are selected according to people's subjective intention, not

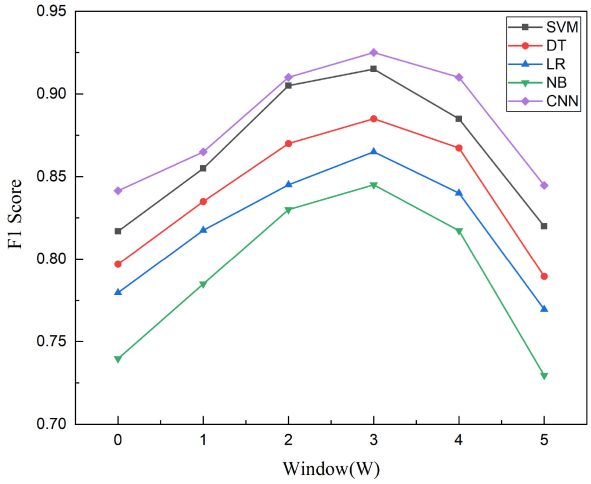

Fig. 10. F1 score comparison with change of window W.

objective. 3) Enumerate features to obtain the relevant semantic features.

The scheme proposed in this article uses the access sequence of URL to construct the input sequence, and then carries out semantic understanding of the sequence information of the current URL to achieve the embedding of the current URL, without human participation in the whole process, which greatly speeds up the efficiency.

Through feature engineering to do ML classification algorithm, the core needs to do feature mining. The more features adopted, the better the classification of the following models and determine the upper limit of the overall classification. Here the lexical features is a kind of mining method, the more characteristics of this type, the more conducive to the overall classification performance.

V. Conclusion

In this article, we introduced a new unsupervised learning algorithm called the UE model, for solving feature subjectivity problems. The UE classifier significantly improved the performance of malicious information identification compared with feature engineering classifiers. The accuracy and recall rate of the UE model are much higher than those of traditional algorithms based on feature engineering, such as the DT, LR, CNN, and SVM algorithms. To obtain a better UE model, we evaluated the model with different dimensions of vectors and context windows, respectively. We explored the appropriate dimension for domains, and chose $N = 100$ as the eventual dimension. We also found out that $W = 3$ was the best parameter for the context information of the current URL.

In the future, we will adapt some sampling methods to train the distributed representation of domains, for a faster convergence speed. Moreover, we will systematically test the influence of the UE model with a sampling method, to determine the detection quality of the system. We also need to build a real system for collaborative learning with device, class, or user-level security (e.g., users share data in unpredictable ways).

References

[1] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," *J. Mach. Learn. Res.*, vol. 7, pp. 2721–2744, 2006.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, 2003.

[3] Z. Cui, F. Xue, X. Cai, Y. Cao, G.-g. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," *IEEE Trans. Ind. Inform.*, vol. 14, no. 7, pp. 3187–3196, Jul. 2018.

[4] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, "Semantics-aware malware detection," in *Proc. IEEE Symp. Secur. Privacy*, 2005, pp. 32–46.

[5] Y. Xu, J. Ren, G. Wang, C. Zhang, J. Yang, and Y. Zhang, "A blockchain-based nonrepudiation network computing service scheme for industrial IoT," *IEEE Trans. Ind. Inform.*, vol. 15, no. 6, pp. 3632–3641, Jun. 2019.

[6] B. Cao, M. Li, L. Zhang, Y. Li, and M. Peng, "How does CSMA/CA affect the performance and security in wireless blockchain networks," *IEEE Trans. Ind. Informat.*, early access, Sep. 25, 2019, doi: 10.1109/TII.2019.2943694.

[7] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Serv. Comput.*, early access, doi: 10.1109/TSC.2019.2953033.

[8] R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "Malware classification with recurrent networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2015, pp. 1916–1920.

[9] A. Graves, *Supervised Sequence Labelling With Recurrent Neural Networks*. New York, NY, USA: Springer, 2012.

[10] Y. Xu, G. Wang, J. Ren, and Y. Zhang, "An adaptive and configurable protection framework against android privilege escalation threats," *Future Gener. Comput. Syst.*, vol. 92, pp. 210–224, 2019.

[11] X. Yan, B. Cui, Y. Xu, P. Shi, and Z. Wang, "A method of information protection for collaborative deep learning under GAN model attack," *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, early access, Sep. 10, 2019, doi: 10.1109/TCBB.2019.2940583.

[12] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, and H. Ye, "Significant permission identification for machine-learning-based android malware detection," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3216–3225, Jul. 2018.

[13] N. Duffield, P. Haffner, B. Krishnamurthy, and H. Ringberg, "Rule-based anomaly detection on IP flows," in *Proc. IEEE Int. Conf. Comput. Commun.*, 2009, pp. 424–432.

[14] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Learning to detect malicious URLs," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–24, 2011.

[15] S. Yadav, A. K. K. Reddy, A. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in *Proc. 10th Assoc. Comput. Machinery's Special Interest Group Data Commun. Conf. Internet Meas.*, 2010, pp. 48–61.

[16] D. Plohmann, K. Yakdan, M. Klatt, J. Bader, and E. Gerhards-Padilla, "A comprehensive measurement study of domain generating malware," in *Proc. 25th USENIX Secur. Symp.*, 2016, pp. 263–278.

[17] S. Schüppen, D. Teubert, P. Herrmann, and U. Meyer, "FANCI: Feature-based automated NXDomain classification and intelligence," in *Proc. 27th USENIX Secur. Symp.*, 2018, pp. 1165–1181.

[18] P. Zhao and S. C. Hoi, "Cost-sensitive online active learning with application to malicious URL detection," in *Proc. 19th Assoc. Comput. Machinery's Special Interest Group Knowl. Discovery Data Mining Int. Conf. Knowl. Discovery Data Mining*, 2013, pp. 919–927.

[19] X. Zhou, W. Liang, I. Kevin, K. Wang, R. Huang, and Q. Jin, "Academic influence aware and multidimensional network analysis for research collaboration navigation based on scholarly big data," *IEEE Trans. Emer. Topics Comput.*, to be published, doi: 10.1109/TETC.2018.2860051.

[20] Y. He, Z. Zhong, S. Krasser, and Y. Tang, "Mining DNS for malicious domain registrations," in *Proc. 6th Int. Conf. Collaborative Comput.: Netw., Appl. Worksharing*, 2010, pp. 1–6.

[21] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surv. Tut.*, vol. 18, no. 2, pp. 1153–1176, Apr./Jun. 2016.

[22] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *J. Netw. Comput. Appl.*, vol. 60, pp. 19–31, 2016.

[23] A. Blum, B. Wardman, T. Solorio, and G. Warner, "Lexical feature based phishing URL detection using online learning," in *Proc. 3rd ACM Workshop Artif. Intell. Secur.*, 2010, pp. 54–60.

[24] D. Marino and M. Manic, "Modeling and planning under uncertainty using deep neural networks," *IEEE Trans. Ind. Inform.*, vol. 15, no. 8, pp. 4442–4454, Aug. 2019.
[25] Y. Zhang, R. Deng, D. Zheng, J. Li, P. Wu, and J. Cao, "Efficient and robust certificateless signature for data crowdsensing in cloud-assisted industrial IoT," *IEEE Trans. Ind. Inform.*, vol. 15, no. 9, pp. 5099–5108, Jun. 2019.
[26] K. Yan, W. Shen, Q. Jin, and H. Lu, "Emerging privacy issues and solutions in cyber-enabled sharing services: From multiple perspectives," *IEEE Access*, vol. 7, pp. 26 031–26 059, 2019.
[27] Y. Xu, Q. Zeng, G. Wang, C. Zhang, J. Ren, and Y. Zhang, "An efficient privacy-enhanced attribute-based access control mechanism," *Concurrency Comput.: Pract. Experience*, vol. 32, no. 5, pp. 1–10, 2020, doi: 10.1002/cpe.5556.
[28] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review," in *Data Clustering*. London, U.K.: Chapman & Hall, 2018, pp. 29–60.
[29] L. Jiang, L. Zhang, C. Li, and J. Wu, "A correlation-based feature weighting filter for naive bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 201–213, Feb. 2019.
[30] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*.

Xiaodan Yan received the B.S. degree in computer science and technology, from Qufu Normal University, Jining, China, in 2013. He is currently working toward the Ph.D. degree with the National Engineering Laboratory of Mobile Internet Security Technology at Beijing University of Posts and Telecommunications, Beijing, China, with a major in network information security.

He is now mainly engaged in network attack and defense technology, vulnerability mining technology, big data security analysis technology, mobile Internet, and Internet of Things security technology research. His research interests include network and systems security, distributed systems, and networking.

He is a member of China Computer Federation (CCF). He serves as the Publicity Chair for International Workshop on Cyberspace Security (IWCSS 2020), and as a Reviewer of over 5 international journals.

Yang Xu (Member, IEEE) received the Ph.D. degree in computer science and technology, from Central South University, Changsha, China, in 2019.

From 2015 to 2017, he was a Visiting Ph.D. Student with Texas A&M University, College Station, TX, USA. He is currently an Assistant Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. He is working in the areas of network computing, blockchain, and operating system. He has authored or coauthored over 30 articles in international journals and conferences, including IEEE TRANSACTIONS ON SERVICES COMPUTING(TSC), IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS (TII), IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS (TCBB), etc.

Dr. Xu was the awardee of the Best Paper Award of the 4th IEEE International Conference on Internet of People. He is a member of CCF Technical Committee on Blockchain. He serves as the Program Committee Chair for IWCSS 2020, the Publicity Chair for the 13th IEEE International Conference on Cyber Physical and Social Computing, and as a Reviewer of over 10 international journals.

Baojiang Cui received the B.S. degree from the Hebei University of Technology, Hongqiao, China, in 1994, the M.S. degree from the Harbin Institute of Technology, Harbin, China, in 1998, both in materials science, and the Ph.D. degree in control theory and control engineering from Naikai University, Tianjin, China, in 2004.

He is currently a Professor with the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China. He has authored or coauthored over 30 articles. His current main research interests include detection of software, cloud computing, and the Internet of Things.

Dr. Cui was the awardee of the Best Paper Award of the 3rd International Conference on Broadband Network & Multimedia Technology and the 9th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing.

Shuhan Zhang is currently an undergraduate student with the National Engineering Laboratory of Mobile Internet Security Technology at Beijing University of Posts and Telecommunications, Beijing China, with the major of Internet of Things Engineering.

Recent years, she has authored or coauthored more than two technical papers in international journals and conferences. She is now mainly engaged in Internet of Things, artificial intelligence technology, mobile computing, network performance analysis, vulnerability mining technology, big data security analysis technology. Her current main research interests include secure data storage, blockchain network architecture and security, network security, mobile social network and mobile healthcare with cryptography and machine learning, and cloud computing.

Taibiao Guo received the bachelor's degree in electronic information engineering, from the China University of Geosciences, Wuhan, China, in 2017.

He has been involved in the development of blockchain for 3 years in the industry. He is now working as a Laboratory Researcher with Blockchain Lab, Hunan University of Technology and Business, Hunan. He is mainly engaged in network attack and defense technology, software and operating system security defect analysis, mobile Internet and Internet of things security technology research. His current research interests include blockchain network architecture and security, secure data storage, secure data computation and user authentication in mobile cloud computing, mobile social network and mobile healthcare with cryptography and machine learning.

Chaoliang Li received the B.S. degree in applied mathematics from HuaiBei Normal University, HuaiBei, China, in 1996, and the M.S. degree in computer application technology, and the Ph.D. degree in computer software and theory from Central South University, Changsha, China, in 2004 and 2015, respectively.

He is currently an Assistant Professor with Hunan University of Technology and Business, Changsha, China. He has authored or coauthored more than 20 technical papers and books in the above areas. His research is supported by the Hunan Provincial Natural Science Foundation, the Hunan Provincial Education Department Scientific Research Foundation. His current research interests include artificial intelligence, big data, Internet of Things, Blockchain, privacy and security issues in IoT.