Accepted Manuscript

Incorporating URL embedding into ensemble clustering to detect web anomalies

Bo Li, Guiqin Yuan, Li Shen, Ruoyi Zhang, Yiyang Yao

PII: S0167-739X(18)32000-4
DOI: https://doi.org/10.1016/j.future.2019.01.004
Reference: FUTURE 4697

To appear in: *Future Generation Computer Systems*

Received date : 19 August 2018
Revised date : 3 December 2018
Accepted date : 1 January 2019

Please cite this article as: B. Li, G. Yuan, L. Shen et al., Incorporating URL embedding into ensemble clustering to detect web anomalies, *Future Generation Computer Systems* (2019), https://doi.org/10.1016/j.future.2019.01.004

Incorporating URL Embedding into Ensemble Clustering to Detect Web Anomalies

Bo Li[a,*], Guiqin Yuan[a], Li Shen[b], Ruoyi Zhang[c], Yiyang Yao[d]

[a]State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
[b]Marketing Department, State Grid Hangzhou Power Supply Company, Hangzhou, China
[c]Regulation and Control Center, State Grid Hangzhou Power Supply Company, Hangzhou, China
[d]State Grid Zhejiang Electric Power Company Information Telecommunication Branch, Hangzhou, China

Abstract

Web anomaly detection aims to find deviations from normal behaviour that happened in our system at most of the time. With the development of the Internet, it is vital for the security of the Internet to detect web-based anomalies. Clustering based on feature extraction by manually has been verified as a significant way to detect new anomalies. But the presentations of these features can't express semantic information of the URLs. In addition, few studies try to cluster the anomalies into specific types like SQL-injection. In order to solve these two problems, we provide a weighted deep learning enabled subspace spectral ensemble clustering approach for web anomaly detection called WDL-SSEC. This approach has three steps. Firstly, an ensemble clustering model is applied to separate anomalies from normal samples. Then we use word2vec to get the semantical presentations of tokens and concatenate weighted tokens to get vectors of the URLs. Finally, another ensemble clustering based on subspace and locally adaptive clustering (LAC) multi-cluster anomalies into specific types. Our approach is run on a real-life data set. The results achieves better performance than existing approaches, which demonstrates that our model has the ability to cluster anomalies into appropriate types.

Keywords: , anomaly detection, weighted deep learning, subspace weighted ensemble clustering

*Corresponding author
Email address: libo@act.buaa.edu.cn (Bo Li)

1. Introduction

Web application has become the most popular applications in the Internet. However, in recent years, web security is fast becoming a big concern for many businesses. According to the Data Breach Report of Verizon, web attack has been ranked as the Top 1 threat among all kinds of security risks.As a result, web applications now need more protections than ever.

Intrusion detection is a useful approach to detect web attacks. Intrusion detection could be divided into two categories: misuse detection and anomaly detection. Traditional misuse detection methods try to detect anomalies by matching signature patterns. For example, Snort [1] has appropriate rules for the web-related attacks. However, all of the patterns are based on the known attacks. The system can't detect anomalies which have not ever appeared in the system.

Anomaly detection [2] narrowly refers to those methods that use machine learning to detect outliers, which includes supervised anomaly detection and unsupervised anomaly detection. Supervised anomaly detection aims to methods which build model based on labeled data. Many traditional supervised machine learning methods can be used to detect anomalies, such as support vector machine, decision tree, random forest. By extracting features manually, they try to build a model based on normal data and anomalous data [3][4]. William Robertson [5] showed that anomaly detection can be done with scarce labeled data. All these studies still need labelled training set and don't have the ability to detect novel anomalies.

Compared to supervised anomaly detection, unsupervised anomaly detection [6][7] has the ability to find new anomalies without labeled data. Moreover, the anomaly detection system without the labeled data or signature pattern, can avoid becoming tedious and error-prone. But the detection rate is lower, and the false positive rate is higher [8][9][10][11], which may cause serious problem in network intrusion detection system. The relatively high cost of high false positive rate makes the methods based on clustering is hard to deploy in real life. The ensemble clustering [12] provides a way to alleviate the defects. Because of this, it can be an implement of existing intrusion detection system. In addition, clustering has a dimension disaster when facing high-

dimension data. To solve this problem, we propose a method based on subspace to cluster anomalies into specific types, such as sql injection. In order to make the data in the same cluster is more close to each other than other data, the subspace ensemble clustering is based on locally weighted clustering[13]. On the other hand, we can get corresponding weights of features in different clusters, which is beneficial to translate our result in real intrusion detection system.

Why URL embedding?. The features extracted by manually can't present the semantic information of URLs. For instance, the character distribution [14] means the count of different printable characters that appeared in a URL. It is a subset of the 256 8-bit values, and order it in descending order or ascending order. For example, "www.google.com" can be represented by $\{3, 3, 2, 1, 1, 1\}$. The character distribution of anomalous data is different from the normal. Such as, the buffer overflow attacks which send binary data can be detected by the attribute. T Threepar [15] showed that we can detect anomalies by entropy. The lower the entropy is, the more likely the URL is an anomalous record. However, all of these features don't encode so much sematic meaning like in natural language processing and loss too much information in URL. Furthermore, the semantics of URLs have been ignored by previous researches.

The existing methods to cope with anomaly detection don't take the semantic information into account. And the systems based on signature and supervised anomaly detection don't have the ability to detect new anomalies. To address these two problems, we provide a method (DL-SWEC) which multi cluster anomalies into specific types by URL embedding and subspace weighted ensemble clustering (SWEC). It is composed of ensemble clustering, URL embedding and SWEC. The development of deep learning [16][17][18][19] gives us a chance to explore more useful information. Therefore, the features based on deep learning are the appropriate choice for us to replace extracting features by manually. In this paper, we have the following contributions:

- We incorporate the word2vec [20][21][22][23] into the multi-clustering model, which can extract semantical information of URLs and needn't specialized knowledge.
- We get efficient URL embedding [24] by word embedding, which can solve the

problem lacking professional stopping words in web anomaly detection.

- We extend the Spectral Ensemble Clustering (SEC) [25][26] to get Subspace Weighted Ensemble Clustering (SWEC) by generating more quality basic partitioning, whose speed is 28 times faster than SEC and cluster each cluster with corresponding features tending to cluster URLs more closely.

Overview. The remainder of this paper is organised like this. In section 2, we present the related work. In section 3, we show some preliminaries and problem definition. In section 4, we show the architecture of our algorithm and present URL embedding algorithm and subspace weighted ensemble clustering. In section 5, we present results and analysis of experiment. In the last section, we make a conclusion of our current work.

2. Related Work

Anomaly detection aims to solve this kind of problem which try to find behaviours acting beyond we expected [9] [2] [27]. In this paper, we mainly refer to point anomalies, neither contextual anomalies nor collective anomalies[2], which means that each individual point can be recognised as abnormal with respect to the rest of data. Unsupervised anomaly detection builds their model based on the premise that the amount of normal data is far more than abnormal data [9].

Compared with supervised anomaly detection, unsupervised anomaly detection what needn't labelled data is more significant and realistic. In order to build an efficient anomaly detection, we need preprocess our data set effectively. Moreover, the most important part in data preprocessing is feature construction and reduction [28]. There are many studies which has make impressive progress in this area [14] [15] [27] [28] [29] [30]. Kruegel C and Vigna G proposed that attribute length, attribute character distribution et al. can be used to detect anomalies with different model in [27]. [29] proposed four types of features, namely redirection and cloaking, deobfuscation, environment preparation, and exploitation. In this paper, they build their model based on JavaScript code and deploy this model in real life which has been used by thousands

of analysts. All these features need researchers have specialised knowledge and it is difficult to describe our data efficiently, especially when facing novel attacks. In our last paper [31], we build URL representations by add vector of tokens that are generated by split the URL with non-alphanumeric characters. We train our data set using word2vec model to get vector of tokens, which is a state-of-art algorithms natural language processing (NLP). In this paper, we try to describe URL more accurately by distributing different token with corresponding weight and removing redundancy part.

There are many unsupervised anomaly detection methods to detect anomalies [8] [9] [10] [11] [32] [33] [34]. Eskin E, Arnold A et al.[8] map input space into Hilbert space firstly, then propose three unsupervised anomaly detection methods called Cluster-Based, K-Nearest and One Class SVM. Cluster-Based and K-Nearest try to detect anomalies based on the premise that anomalies has less compact neighbors in contrast to normal behaviour. And One Class SVM separate the mapped data from origin by hyperplane. Zhang J and Zulkernine M [9] transforms unsupervised anomaly detection into a supervised anomaly detection by the type of network service. They build a model called random forest based on labelled network service type and judge whether a record is anomaly or not by the result produced by the model is according to the original network type. Unfortunately, as mentioned in our last paper [31], unsupervised anomaly detection system faces a serious problem that the result of model is unstable and has high false positive rate. In this paper, we also use ensemble clustering to solve this problem. In addition, after distinguishing between normal and abnormal by ensemble clustering, we use another ensemble clustering based on subspace to cluster anomalies into specific types for solving the curse of dimension, which is helpful for researchers to translate our result. And by using locally weighted clustering to build basic partitioning for ensemble clustering, we can choose different features for different clusters. It is reasonable to do like this, because different clusters have different structures, such as SQL injection, XSS-nonpersistent.

3. Preliminaries

Here we briefly introduce some basic algorithms, which will be used in our next partition.

One Class SVM. One class SVM [35]originate Support Vector Machine (SVM), which is an efficient supervised machine learning algorithm. SVM solve the classification problem by finding the maximum separation interval hyperplane, which is a quadratic optimization problem. One-class SVM is a natural extension of SVM in unlabelled data. Given a data set X, which includes n point in it, and each point is represented in a d dimensional vector. We use a kernel function to map origin data into a high dimensional space, and separate them from the origin with maximum interval.

We solve this problem as follows,

$$\min_{w,\xi,\rho} \frac{1}{2} \| w^2 \| + \frac{1}{vd} \sum_i \xi_i - \rho \tag{1}$$

$$subject\ to\ (w.\varphi(x_i)) \geq \rho - \xi_i \tag{2}$$

$$\xi_i \geq 0 \tag{3}$$

where $\phi(x_i)$ is the kernel function, which map the origin data into a high dimensional space. And ξ_i is non zero slack variable, which is penalised in our objective function. w is the presentation of our hyperplane. So our decision function is defined as follows,

$$f(x) = sgn(w \cdot \phi(x) - \rho) \tag{4}$$

If the result of $f(x)$ is +1, we judge the data is normal, otherwise is an abnormal data. And we can understand the algorithm in another way, we try to find a suprasphere to circle all our data. And if the data is in the suprasphere, we judge the data is normal, otherwise it is an anomaly data, which is been testified in [35].

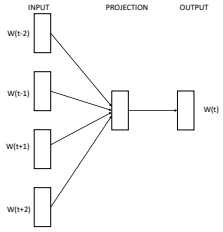


Figure 1: Architecture of word2vec based on CBOM

Gaussian Mixture Model. Gaussian Mixture Model (GMM) [36] is one of most important application of Expectation Maximum (EM). I mean this kind of distribution like follows,

$$P(Y = y|\theta) = \sum_{k=1}^{k} \alpha_k \phi(y|\theta_k) \tag{5}$$

The distribution of Y is multiple mixture of gaussian model, θ_k stands for the parameter of the Kth gaussian distribution, and α_k is the weight of the Kth gaussian distribution, which subject to the following formulation

$$\phi(y|\theta_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} exp(-\frac{(y-\mu_k^2)}{2\sigma_k^2}) \tag{6}$$

By solving the following maximum likelihood problem by EM, we can get the GMM, which is used in unsupervised anomaly detection extensively.

$$L(y|\theta) = \prod_{i=1}^{N} P(y_i|\theta) \tag{7}$$

In this paper, we will use One class SVM and GMM to consist of the basic partitioning of ensemble clustering by cluster the original data set into normal set and abnormal set, so we briefly introduce these two algorithm here.

Word2vec. In this section, we briefly introduce the CBOW [22] model based on Negative sampling firstly, which is one of the implementation for word2vec. And we show the architecture of CBOW in the following,

The CBOW model tries to predict the word (w_t) based on the known words ($w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$) when *window* = 2. The subscript represents the position of the word appears in the context. It's worth noting that $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ are from the set of negative sampling, while not completely from the two words before and after the current word. The Negative Sampling framework can be implemented as follows. In this framework, it tries to decrease the weight of high-frequency words, while increase the weight of low-frequency words. In this way, this model can get better word representation by removing redundant words like com, www, etc.

The presentation of current word can be generated during the process of maximize the following equation (8).

$$P(w|Context(w)) = \prod_{u \in w \cup NEG(w)} g(w) \tag{8}$$

$$g(w) = [\sigma(x_w^T \theta^u)]^{L^w(u)} [1 - \sigma(x_w^T \theta^u)]^{1-L^w(u)} \tag{9}$$

$$\sigma(x) = \frac{1}{1 + e^{-x^T \theta}} \tag{10}$$

In equation (8)(9)(10), $NEG(w)$ denotes the set of negative samples. $L^w(u)$ means the label of w when the current word is u. If $w = u$, $L^w(u) = 1$, else $L^w(u) = 0$. θ^u stands for the parameters of the current word in sigmoid model. The $Context(w)$ consists of several words before and after u. Most importantly, x_w means the vector of current words, which is initialized by the sum of vectors of $Context(w)$. The vector will be updated in the process of maximum the equation (8). In particular, the equation (8) can be transformed to (12). From equation (12), it's obvious that the aim of maximum equation (8) is to maximize the probability of positive samples while minimize the probability of negative samples. For a given training set C, the model need to obtain the maximum of equation (11) by the Stochastic gradient ascent model. The updated process is showed in the following equation (13) and equation (14). In both of them, there is a parameter call η, which means the learning rate in the process of training. And the $v(\vec{w})$ is the vector of the current word. It means that the representation of

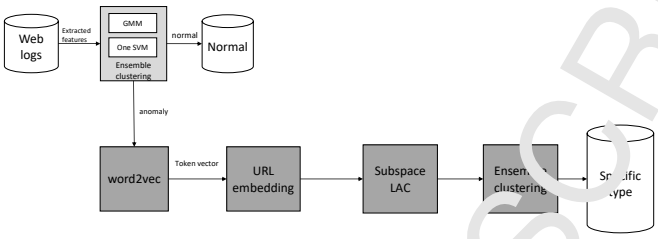


Figure 2: Architecture of multi-clustering model based on DEP-SSEC

current word can be achieved by maximizing equation (8).

$$U = \prod_{w \in C} g(w) \tag{11}$$

$$P(w|Context(w)) = \sigma(x_w^T \theta^w) \prod_{u \in NEG(w)} (1 - \sigma(x_w^T \theta^u)) \tag{12}$$

$$\theta^u := \theta^u + \eta[L^w(u) - \sigma(x_w^T \theta^u)]x_w \tag{13}$$

$$v(w) := v(\vec{w}) + \eta \sum_{u \in w \cup NEGw} [L^w(u) - \sigma(x_w^T \theta^u)]\theta^u \tag{14}$$

Word2vec is a state-of-art tool in NLP, which has got widespread apply in real life. It can provide a distribute representation of word compared with word representation generate by one-hot and n-gram. The distributed vector of word generated by word2vec can make some interesting algebraic operation, such as v(queen)-v(woman) = v(king)-v(man).

4. Algorithm

In this section, we describe the framework of multi-clustering model based on WDL-SSEC. Its architecture is shown in Figure 2. The two models (GMM, one-SVM) circled by a big rectangle constitute Ensemble clustering model, which have the ability to detect new anomalies. GMM and one-SVM produce high quality basic partitioning, and the ensemble clustering based on voting get higher true positive rate, lower false positive rate and more stable result than basic partitioning. The ensemble clustering based on voting chooses the result that the result consistent with majority of basic partitioning, and aims to generate a better clustering result than best basic partitioning.

After that, if a record is normal, it will be stored in the database of normal logs. If a record is anomalous, it will be passed to word2vec model. The task of word2vec model is to generate semantic vectors of tokens, which are generated by split URL with non-alphanumeric character. In particular, the word2vec model will update the model once in a while, while needn't to retrain once anyone of URL arrives. The model can be stored in memory and can be accessed whenever it is necessary. The vectors of words are transferred into URL embedding model, which will produce the efficient presentation of URL. After that, locally adaptive clustering(LAC) based on subspace use these representations of URL to generate basic partitioning. Finally, all these results of basic partitioning will be delivered to ensemble clustering. The task of this component is to multi-cluster the detected anomalies into specific types. Next the URL embedding, and SWEC which is composed of subspace LAC and ensemble clustering will be discussed in detail.

URL embedding. In order to take the semantic information into account, we employ word2vec to present URLs. In the following section, we will introduce how to get the vectors of URLs for the vectors of tokens. Algorithm 1 briefly introduces the process of implementation. The input of the algorithm is the token embedding *v_set* which derived from word2vec. Probability vector *p_set* which derived from the frequency of token divide the sum of all tokens's frequency, which can be calculated from similar intrusion detection system. Scalar a is a constant number to avoid the denominator to be zero. What't more, it is an scalar to adjust the influence of weight of token, if we

set a with a large number, it means that we ignore the weight of token. The output of the algorithm is the vectors of URLs. From line 5 to line 16, the sematic representation of URLs are calculated with weighted word embedding. In particularly, the number of tokens is not equal to the size of *v_set*. It means that the token may not have the corresponding vector.In particularly, the number of tokens is not equal to the size of *v_set*. It means that the token may not have the corresponding vector. After that, for each url embedding, we subtract the main component of main singular vector, which is beneficial to erase redundancy part of each URL embedding.

Algorithm 1 URL embedding

1: INPUT: token embedding *v_set*, probability vector *p_set*, URLs, scalar *a*
2: OUTPUT: URL embedding
3: begin
4: $result \leftarrow EMPTY$
5: **for** $line \in datalist$ **do**
6: $count \leftarrow 0$
7: $e \leftarrow 0$
8: **for** $item \in line$ **do**
9: **if** $item \in v\_set$ **then**
10: $e \leftarrow e + \frac{a}{a+p\_set(item)} \cdot v\_set[item]$
11: $count \leftarrow count + 1$
12: **end if**
13: $e \leftarrow \frac{e}{count}$
14: ADD the e to result
15: **end for**
16: **end for**
17: Run PCA on result to get the first singular vector μ
18: **if** The length of μ is smaller than the number of URL N **then**
19: Filled μ with zero until the length of it equal to N
20: **end if**
21: **for** v(URL) in result **do**
22: v(URL) $\leftarrow$ v(URL)-$\mu\mu^T$ v(URL)
23: **end for**
24: Use Normalizer with $l2$ and StandardScaler to normalise result
25: return result

In order to get word embedding, the model we adopted for word2vec is CBOW model based on Negative sampling. The Negative sampling framework will do sub-sampling, which tries to increase the weight of low-frequency words and decrease the weight of high-frequency words. Another advantage of this strategy is that Negative

Sampling is faster than the Hierarchical Softmax which is the other important framework in word2vec. Compared to Skip-gram model, the CBOW model can run faster and take frequent words into account more. However Negative sampling can make a fine balance between the low-frequency words and high-frequency word by subsampling. This is also the reason why we choose the CBOW model based on Negative sampling. In addition, a URL can be treated as a meaningful sentence, rather than a set of confused characters. It has its own grammatical structure like English. For example, we can use "Google" to express the name of a well-known company in URLs. In NLP, before employ algorithm on data set, we need to preprocess our data set. After tokenize training set, researcher need to preprocess the data set with stopping vocabulary table, in order to wipe off some meaningless token. Unfortunately, there is no professional or common vocabulary table in intrusion detection system. To fix this problem, algorithm 1 distributes high weight to low frequency words, while low weight to high frequency words. [24] refer to this algorithm as smooth inverse frequency (SIF), and testified reweighting of word vectors using generative model for sentences. Furthermore, in line 17 to line 23, algorithm 1 removes the projections of each URL vectors on their first principle component, which is beneficial to keep efficient information for multi clustering.

We observe that this model is an unsupervised way to generate meaningful representation of URLs. What we need to do is to split the URLs into tokens then gather the tokens into a set which will be used as the training set in word2vec model. The Algorithm 1 shows the corresponding process of generating vectors of URLs with the training set. This is a new way to present URLs in web anomaly detection, which is ignored before and an efficient way to present the meaning of URL.

SWEC. In [37], SEC has been proved to be an efficient algorithm in ensemble clustering. In this paper, we extend the algorithm by modifying the methods of generating basic partitioning. In order to introduce the algorithm, we give the following concepts.

A basic partitioning is the result after one clustering was run on the data set. The n means the size of data set, r stands for the number of partitioning, K_i means the number of clusters in the kth partitioning. All of these partitioning constitute the set

$\prod = \pi_1, \pi_2, \ldots, \pi_r$. $B = \{b(x)\ (15)\}$ is the set of binary matrix derived from basic partitioning.

$$b(x) =< b(x)_1, b(x)_2, \ldots, b(x)_r > \tag{15}$$

$$b(x)_i =< b(x)_{i1}, b(x)_{i2}, \ldots, b(x)_{iK_i} > \tag{16}$$

$$b(x)_{ij} = \begin{cases} 1 & \text{if } \pi_i(x) = j, \\ 0 & \text{if } \pi_i(x) \neq j, \end{cases} \tag{17}$$

Based on this, the spectral ensemble clustering can be solved by weighted K-means. It's worth note that original spectral ensemble clustering need to construct the co-association matrix and calculate top k that is cluster number eigenvectors, which is time-consuming and time complexity is $O(n^3)$. And the space complexity is $O(n^2)$. But after the transform, the time complexity of the algorithm decreased from $O(n^3)$ to roughly $O(InrK)$. The I stands the number of iterations of this model and $K = \sum_{i=1}^{r} K_i$. And the generalizability has an upper bound which can be defined like equation (18), which is also the generalizability of SWEC.

Generalizability in [37]. Let π be any one basic partitioning of SWEC. And $x_1, x_2, \ldots, x_n$ are independent from each other. Let $\sigma > 0$, with probability at least $1 - \sigma$:

$$\begin{aligned} E_x f_{m1,\ldots,mk}(x) - \frac{1}{n}\sum_{i=1} f_{m1,\ldots,mk}(x_i) \\ \leq \frac{\sqrt{2\pi}r}{n}\Big(\sum_{i=1}^{n} w_{b(x_i)}^{-2}\Big)^{\frac{1}{2}} + \frac{\sqrt{8\pi}rK}{\sqrt{n}min_{x\in\{x_1,\ldots,x_n\}} w_{b(x)}} \\ + \frac{\sqrt{2\pi}rK}{nmin_{x\in\{x_1,\ldots,x_n\}} w_{b(x)}^2}\Big(\sum_{i=1}^{n} w_{b(xi)}^2\Big)^{\frac{1}{2}} + \Big(\frac{\ln(1/\sigma))}{2n}\Big)^{\frac{1}{2}} \end{aligned} \tag{18}$$

$$f_{m_1,m_2,\ldots,m_n}(x) = min_k w_{b(x)} \| \frac{b_{(x)}}{w_{b_{(x)}}} - m_k \|^2 \tag{19}$$

$$m_k = \frac{\sum_{x \in C_k} b(x)}{\sum_{x \in C_k} w_{b(x)}} \tag{20}$$

Algorithm 2 shows the process of the basic partitioning is constructed in SWEC. It uses the vectors from Algorithm 1 and output the clustering results. In this algorithm, it generates a random number *num*, which represents that features are split into *num* different sets. Then the data in every set is discarded randomly to generate the basic partitioning. In particular, the Data is different in each iteration. We shuffle the features in order to get best subset of features. It also can increase the diversity of basic partitioning. Moreover, this model selects a subset of data not only features, but also data sets. In addition, the basic partitioning built in the way will speed up the clustering. In [38], the author testified that the ensemble classifier will have better performance with diversity and high-quality basic partitioning. This is also appropriate for ensemble clustering. This is also the reason why we build the basic partitioning in this way. From line 21 to line 25, running LAC with h consists of basic partitions. h stands for coefficient that balance the loss and regularization, which controls relative weight of different features. For example, if h is set to infinity, LAC will distribute all weights to unit feature, while zero weight on other features. What's more, LAC can output the corresponding weights for different clusters, which is significant for researchers to translate the output of LAC. In this step, SWEC multi cluster anomalies into specific types, such as SQL injection, XSS-nonpersistent. SQL injection mainly use sql to steal database information, while XSS-nonpersistent mainly use JAVASCRIPT to convey their attempt. That is to say, different cluster probably employ different feature to put anomalies gather together.

Now, we have built the WDL-SWEC model. The SWEC has a lower time complexity $O(n)$ compared with traditional ensemble clustering, the ability to do non-linear clustering and a better robustness.

5 Experimental evaluations

Firstly, we verify whether the ensemble clustering has the ability to find anomalies which are ignored by misuse detection or not. Then, we study the effectiveness of

Algorithm 2 Basic partitioning in SWEC

```
INPUT: Data
OUTPUT: Result
begin
for iter=1:10 do
    num ← random number in [0:50]
    shuffle the collum of Feature
    for i in xrange(len(Data[0])) do
        if i + num < len(Data[0]) then
            ADD the Data[i : i + num] to Set
        else
            ADD the Data[i :] to Set
        end if
        i ← i + m
    end for
    for item in Set do
        throw_set ← 1000 random numbers
        for i in throw_set do
            item[i] ← [num zeros numbers]
        end for
    end for
    for item in Set do
        Generate coefficient h randomly
        middle ← run DAC with h on item
        ADD middle to Partitioning
    end for
    Ensemble clustering
end for
return result
```

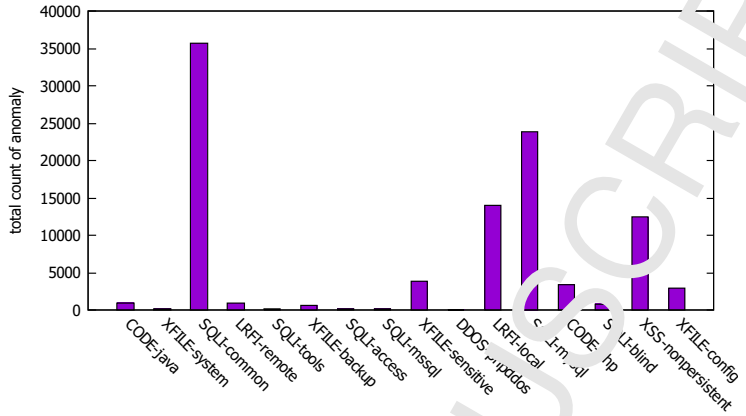


Figure 3: dataset-distributi

features based on URL embedding in comparison with the extracted features by manually, and URL features generated by sum of word embedding. Finally, we evaluate the efficiency of WDL-SWEC in comparison with SEC [37] and the ability of clustering anomalies into appropriate types. All of the models are run on the same computer. It has an Inter(R) Core(TM) *i*5 − 4590 CPU @ 3.30 GHz processor and 8GB memory, running a 64 bit Windows 10 system.

Dataset. All of our experiments were run on a real-life data. The dataset was collected by Qihoo 360, which is a famous company in the field of security. There are many kinds of anomalies in it. But in our experiment, we randomly choose a subset from the original dataset. In fact, the rate of normal data in real life is over 90%. In order to make our dataset more realistic, our dataset consists of 1.6% anomalies and 98.4% normal records. Moreover, we chose the six most anomalies which are chosen randomly. The distribution of anomalies can be viewed in Figure 3. In Figure 3, the y axis stands for the number of the each anomaly appeared in the original data set. And the names of anomalies were arranged on the x axis. Therefore, this paper selects six kinds of anomalies. They are SQLI-common, SQLI-mysql, XSS-nonpersistent, CODE-php, LRFI-local, XFILE-sensitive.

Ensemble clustering. In this section, we try to test if the ensemble clustering can detect the anomalies which were ignored by the signature based intrusion detection system. In order to verify our hypothesis more scrupulously, all models use the high-quality features which were testified by the researchers [14][39][40]. The ensemble clustering based on voting is compared to traditional GMM model, KNN model, One-class SVM model and Cluster model. In [8], GMM model, KNN model and one-class SVM were studied and compared to each other. The clusters of all model was set to 2, which means that one is anomalous, while the other is normal. All of the comparisons were run 10 times in the same environment, and all of models are carried out on the dataset showed in Table 2. The ensemble models ensemble the results of GMM model and one-SVM model. In order to get the basic partitioning, we run GMM model 20 times and one-SVM model 20 times. Table 1 shows the results of different methods on detection rate, false positive rate. From Table 1, we can find that our ensemble clustering achieves a True Positive Rate (TPR) of 92.3% and a False Positive Rate (FPR) of 0.37%. The reason is that GMM is a probabilistic model, while one-SVM is a classifier model. A probabilistic model has excellent capability of data description and one-SVM has excellent data classifier ability. The ensemble clustering can combine the advantages of them, and alleviate the defects at the same time. Moreover, from Table 1, we can conclude that the performance of ensemble clustering is stable and effective.

In addition, our unsupervised ensemble clustering can detect anomalies which weren't detected by the anomaly detection based on signature. In our dataset, the first 600 web logs are anomalies, and the rest of it are normal records. The distribution of anomalies is the same as the last part. The Table 3 shows a subset of anomalies detected by the ensemble clustering, while wasn't detected by misuse detection. The table shows the index of the anomalies appeared in our experimental dataset and the decoded URL. For example, the 5665*th* record is a SQL-mysql injection. The "1 = 1″ is true forever. And the "information_schema" is the system table of Mysql and contains all information for the tables. The hackers try to get the table of our system by repeating testing. The 10966*th* record is a CODE-php Trojan. It tries to generate a Trojan by "xiaosi.php". The detection process always act like following. First the hacker post a URL like the 10966*th* record. Then this record can insert one record into our

Table 1: Selected points from the ROC curves of each algorithm

Algorithm	Detection rate	False positive rate
KNN	85.66%	0.[illegible]5%
KNN	86%	0.15%
KNN	85.33%	0.[illegible]6%
GMM	83.16%	0.07%
GMM	83.5%	0.07%
GMM	88.66%	0.11%
one-SVM	34.6%	0.62%
one-SVM	67.16%	2.28%
one-SVM	91.5%	7%
Cluster	72%	0.36%
Cluster	87.[illegible]%	1.11%
Cluster	91.6[illegible]%	1.6%
ensemble clustering	91.5%	0.2%
ensemble clustering	92.3[illegible]	0.37%
ensemble clustering	91.[illegible]%	0.35%

Table 2: dataset

Norma[illegible]	54000
SQL[illegible]i-comm[illegible]n	100
SQLI-[illegible]ysql	100
XSS[illegible]onpersistent	100
CODE-php	100
LRFI-local	100
XFIL[illegible]-sensitive	100

Table 3: Records detected by our ensemble clustering

index	record
5665	/item.php?act=search&keyword=?' and(select from(select count(*),concat ((select (select (SELECT distinct concat(0x7e,0x27,char(99,102,114,101,101,114), 0x27,code) FROM information_schema.schemata LIMIT 0, 1)) from information_schema.tables limit 0,1),floor(rand(0)*2))x from information_schema.tables group by x)a) and 1 = 1#&searchsort=subject&catid=0&ordersort=adddtime&ordertype=asc&searchsubmit=yes 493
10966	cfg_dbprefixmytag‘ (aid,expbody,normbody) VALUES(9527,@‘\’‘,’dede:phpfile_put_content(”diaosi.php”,” ¡?php eval($_POST[diaosi]);?¿”);/dede:php’) # @‘\’‘
27604	cfg_dbprefixmyad‘ (aid,normbody) VALUES(10002,’¡?php echo ”dedecms 5.7 0 day¡br¿tj, tj”;@preg_replace(”/[copyright]/e”,$_REQUEST[”tongji”], ”error”);?¿’);

database. Finally the hacker can use tools to connect our database. The 27604*th* record is a CODE-php injection. The result shows that our ensemble clustering model has the ability to detect the anomalies, which wasn't recognized by misuse detection, because it's impossible for misuse detection system to extract all patterns for anomalies and build rules for unknown anomalies. But the ensemble clustering has the ability to detect unknown anomalies with unlabeled data.

Multi-clustering based on deep learning. The aim of the experiments in this part is to testify the validity of features based on URL embedding in clustering the anomalies into specific types. And the baseline is the features like attribute length used by classical unsupervised anomaly detection. Furthermore, these features also are valid for anomaly detection, which has been verified in the last section. Moreover, in order to testify the validity of features extracted by URL embedding, we test them on three different clustering methods, which called MiniBatchKMeans, KMeans Consensus Clustering and Spectral ensemble clustering. The set of anomalies in the last section seems to be too small for this task, so we rerun the last section with a data set which is ten times that of last section. We assume that if the model can work on a large dataset, it would be effective in the small dataset. In clustering, we set $n\_cluster = 5$, because we treat SQL-common and SQL-mysql as one type for their similar structure, while keeping the

Table 4: Dataset for multi-cluster

SQLI-common	1000
SQLI-mysql	1000
XSS-nonpersistent	600
CODE-php	1000
LRFI-local	1000
XFILE-sensitive	1000

Table 5: Results of multi-cluster MiniBatchKmeans

	index	**SQL**	**XSS**	**CODE-php**	**LRFI-local**	**XFILE-sensitive**
WDL	TPR(%)	95.2	93.8	81.4	92.6	92.4
	FPR(%)	0.875	0.86	0.56	6.46	1.1
Word2vec	TPR(%)	94.85	91.3	35.5	95.8	90.4
	FPR(%)	0.5	0.7	0.54	17	0.6
Traditional	TPR(%)	33.7	0.4	0	0	100
	FPR(%)	0	7.6	1.28	11.92	60.32

types of remaining anomalies unaltered. In this section, all models are run ten times in order to avoid the disturbance of instability.

In order to get the features expressed by semantic vector, the url embedding model was trained on the result showed in Table 4. And the parameters of word2vec are set as follows, *size* = 400, *window* = 4, etc. *size* parameter means the dimension of feature, and *window* parameter means that we will take four words before and after the current word into account. After get the word embedding, we employ the URL embedding on our data set, which can get weighted sum of word embedding and remove redundant part that URL embedding is mapped to the maximum singular vector. Table 5,6,7 show the comparison of MiniBatchKMeans model, KMeans Consensus Clustering model and Spectral Ensemble Clustering model. The *URL embedding* label means that the

Table 6: Results of multi-cluster Kmeans Consensus clustering

	index	**SQL**	**XSS**	**CODE-php**	**LRFI-local**	**XFILE-sensitive**
WDL	TPR(%)	94.9	93.5	95.3	85.8	89.0
	FPR(%)	0.45	0.86	4.82	3	0.28
word2vec	TPR(%)	65.8	75.6	54.8	42.2	78.6
	FPR(%)	0.15	0.86	11.14	7.92	23.4
Traditional	TPR(%)	55	21.7	70.4	78.2	100
	FPR(%)	0	17.8	15.66	5.92	4.36

Table 7: Results of multi-cluster Spectral Ensemble clustering

	index	SQL	XSS	CODE-php	LRFI-loca	XFILE-sensitive
WDL+WEC	TPR(%)	94.65	93.3	74.4	99	95.0
	FPR(%)	0.15	0.68	0.2	8 8	0.04
WDL	TPR(%)	94.55	93.9	83.7	97.	75.8
	FPR(%)	0.325	1.26	0.2	10.3	0.04
Word2vec	TPR(%)	95.55	92.8	35.2	97.6	90.4
	FPR(%)	0.475	0.84	0.16	17.02	0.18
Traditional	TPR(%)	78.55	100	30.7	67.7	100
	FPR(%)	0	14.44	0	8	6.46

Table 8: Multi-clustering comparison with NMI

Algorithm	MBKmeans	FCC	SEC
URL embedding	0.79	0.8050	0.7828
word2vec	0.71	0.7 4	0.7418
traditional	0.39	0.69	0.7056

feature used by model is based on URL embedding. The *word2vec* label means that the feature used by model is based on word2vec. The *traditional* label means that the feature used by model is extracted by manually. The result shows that the features based on word2vec have the ability to multi-cluster detected anomalies into specific types. In addition, compared to traditional features, the features based on deep learning have better performance in multi-clustering. Table 5 shows the comparison of TPR and FPR in two feature sets on MiniBatchKMeans. From the table, we can see that word2vec has better performance in all types. Although traditional features have better results in XFILE-sensitive in TPR, it has a bad FPR of 60.32%. Table 6 can draw a similar conclusion like this that the multi-clustering based on traditional feature has a higher FPR. From Table 7, the FPR of traditional features nearly all is over 4%. The reason is that the features based on word2vec can express semantical information of URLs, while the traditional features just can separate the normal from anomaly not to multi-cluster. The results indicate that features based on word2vec have better ability to multi-cluster anomalies into specific types.

From table 5, 6, 7, we can see that all these algorithm based on URL embedding show better performance that other features. In table 5, MiniBatchKMeans based on URL embedding has a pretty good TPR of 81.4%, which is much better than other two

ways to get features. Moreover, the FPR in all types is less than 10%, which is meaningful in real life. And all these algorithm based on URL embedding also show better performance that result has a higher TPR and a lower FPR. The higher TPR is, we can recognise more true anomalies, which means that this algorithm can detect more attacks which is true attack in real life. The lower FPR is, this algorithm is more reliable, which means that less normal behaviour is recognised as anomaly in our intrusion detection system. In table 8, it also shows that algorithm based on URL embedding has a higher NMI than these algorithm based on word2vec and traditional features. We can see that features based on word2vec and URL embedding have the ability to describe the structure of different type of anomalies, which is beneficial to multi cluster anomaly into specific types. In addition, algorithm based on URL embedding shows a better performance that the other, the reason is that URL embedding employ SIF algorithm to get URL vector, which can decrease the weight of high frequency words that have less information, while increase the weight of low frequency words that improve the probability of that the URL may take virus with it. As we all knows, hackers always want to conceal their real intension by sophisticated encryption schemes to escape our intrusion detection system, which may generate unusual tokens like normal behaviour.

In summary, algorithm based on word2vec and URL embedding show better performance than these algorithms based on traditional features. These features have the ability to cluster the original data set into normal set and abnormal set, while they aren't designed to multi cluster data set into different specific type, such as SQL, XSS-nonpersistent. Features based on word2vec and URL embedding have a better ability to describe the semantic of URL, which is helpful for analysing different structure of different specific anomaly type. SQL injection based on SQL and XSS-nonpersistent based on JAVASCRIPT are two different kinds of anomaly, while SQL and JAVASCRIPT are two different programming language. And features based on URL embedding can alleviate the influence of redundant tokens at the next level.

VDL-SWEC. Finally, we evaluate the performance of multi-clustering in comparison with DEPSSEC, SEC [37], KCC, MiniBatchKMeans. The five models are also run on the last section's data set and try to cluster the extracted anomalies into specific types.

Table 9: Results of subspace spectral ensemble clustering based on deep learning

	index	SQL	XSS	CODE-php	LRFI-loca	XPATH-sensitive
SSEC	TPR(%)	96.25	93.8	88.2	97.6	90.2
	FPR(%)	0.4	1.0	0.54	5.44	0.26
WDL+SSEC	TPR(%)	95.55	93.4	90.2	95.2	90.8
	FPR(%)	0.45	0.6	1.58	4.92	0.24
WDL+SWEC	TPR(%)	95.4	93.2	94.5	95.9	92.6
	FPR(%)	0.525	0.66	0.98	4.12	0.42

Table 10: Comparison with Rn and NMI

Algorithm	Rn	NMI	time
SEC	0.7776	0.7946	3*h.* 6′
DEP-SSEC	0.8691	0.832	7′ 1″
WDL-SSEC	0.8517	0.817	7′21″
WDL-SWEC	0.8781	0.8285	1′12″

The number of clusters is all set to $[1 : \sqrt{n}]$. The number of basic partitionings of each time in SEC, DEP-SEC and DEP-SSEC, WDL-SWEC is 100. Table 10 shows the results of comparison, which includes the results of Rand Index (Rn), NMI and running time. We can see our model and DEP-SSEC has better performance of multi-clustering than SEC. The speed of our algorithm is nearly 28 times faster than the original, Rn and NMI is higher than DEP-SEC. In Figure 4 and Figure 5, *MBKMeans* stands for MiniBatchKMeans. Table 4 and table 5 show that our WDL-SWEC show better stability and robustness. What't more, WDL-SWEC can output corresponding weights of features for different clusters when consists of basic partitioning, which is meaningful for researchers to understand our results. The result shows that our model has the best performance of multi-clustering in all models. The reason is that the basic partitioning in DEP-SSEC and WDL-SWEC is more various and more quality, and our feature can present the semantic information of URLs.

6. Conclusion

We propose the WDL-SWEC model. We compare the features based on word2vec with the features extracted by manually. By enumerating the anomalies which aren't recognized by misuse detection, it shows that our system can be a significant part of in-

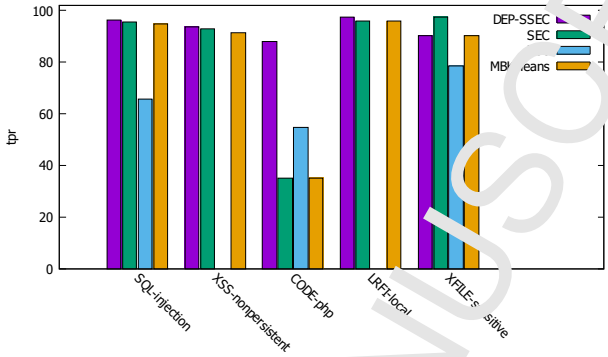


Figure 4: TPR Comparison of all models

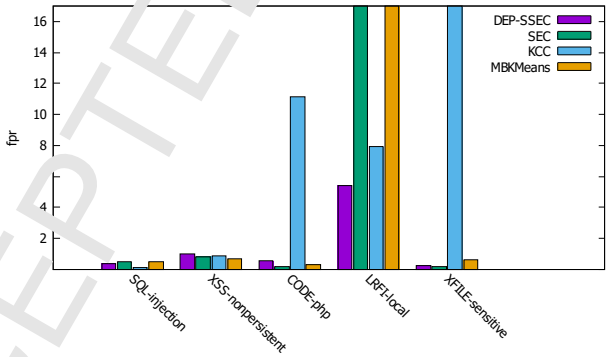


Figure 5: FPR Comparison of all models

trusion detection system. And our WDL-SWEC model has the ability to multi cluster the extracted anomalies into specific types with the best performance. Moreover, our model is 28 times faster than SEC. In particular, our model is an integrated unsupervised anomaly detection system, which can detect anomalies and cluster them into specific types. In the future, we try to build a model based on deep learning from the feature selection to anomaly detection.

References

[1] M. Roesch, et al., Snort: Lightweight intrusion detection for networks., in: Lisa, Vol. 99, 1999, pp. 229–238.

[2] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM computing surveys (CSUR) 41 (3) (2009) 15.

[3] A. Este, F. Gringoli, L. Salgarelli, Support vector machines for tcp traffic classification, Computer Networks 53 (14) (2009) 2476–2490.

[4] Y. Wang, A multinomial logistic regression modeling approach for anomaly intrusion detection, Computers & Security 24 (8) (2005) 662–674.

[5] W. K. Robertson, F. Maggi, C. Kruegel, G. Vigna, Effective anomaly detection with scarce training data., in: NDSS, 2010.

[6] M. Yousefnezhad, D. Zhang, Weighted spectral cluster ensemble, in: Data Mining (ICDM), 2015 IEEE International Conference on, IEEE, 2015, pp. 549–558.

[7] A. K. Jain, Data clustering: 50 years beyond k-means, Pattern recognition letters 31 (8) (2010) 651–666.

[8] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, S. Stolfo, A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data, Applications of data mining in computer security 6 (2002) 77–102.

[9] J. Zhang, M. Zulkernine, Anomaly based network intrusion detection with unsupervised outlier detection, in: Communications, 2006. ICC'06. IEEE International Conference on, Vol. 5, IEEE, 2006, pp. 2388–2393.

[10] L. Portnoy, E. Eskin, S. Stolfo, Intrusion detection with unlabeled data using clustering, in: In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001, Citeseer, 2001.

[11] P. Casas, J. Mazel, P. Owezarski, Unsupervised network intrusion detection systems: Detecting the unknown without knowledge, Computer Communications 35 (7) (2012) 772–783.

[12] S. Vega-Pons, J. Ruiz-Shulcloper, A survey of clustering ensemble algorithms, International Journal of Pattern Recognition and Artificial Intelligence 25 (03) (2011) 337–372.

[13] C. Domeniconi, D. Gunopulos, S. Ma, B. Yan, M. Al-Razgan, D. Papadopoulos, Locally adaptive metrics for clustering high dimensional data, Data Mining and Knowledge Discovery 14 (1) (2007) 63–97.

[14] C. Kruegel, G. Vigna, W. Robertson, A multi-model approach to the detection of web-based attacks, Computer Networks 48 (5) (2005) 717–738.

[15] T. Threepak, A. Watcharapupong, Web attack detection using entropy-based analysis, in: Information Networking (ICOIN), 2014 International Conference on, IEEE, 2014, pp. 244–247.

[16] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[17] Y. Kim, Y. Jernite, D. Sontag, A. M. Rush, Character-aware neural language models., in: AAAI, 2016, pp. 2741–2749.

[18] R. K. Srivastava, K. Greff, J. Schmidhuber, Training very deep networks, in: Advances in neural information processing systems, 2015, pp. 2377–2385.

[19] N. Srivastava, E. Mansimov, R. Salakhudinov, Unsupervised learning of video representations using lstms, in: International Conference on Machine Learning, 2015, pp. 843–852.

[20] M. U. Gutmann, A. Hyvärinen, Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics, Journal of Machine Learning Research 13 (Feb) (2012) 307–361.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in neural information processing systems, 2013, pp. 3111–3119.

[22] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781.

[23] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014, pp. 1188–1196.

[24] S. Arora, Y. Liang, T. Ma, A simple but tough-to-beat baseline for sentence embeddings.

[25] A. Y. Ng, M. I. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: Advances in neural information processing systems, 2002, pp. 849–856.

[26] J. Wu, H. Liu, H. Xiong, J. Cao, J. Chen, K-means-based consensus clustering: A unified view, IEEE Transactions on Knowledge and Data Engineering 27 (1) (2015) 155–169.

[27] C. Kruegel, G. Vigna, Anomaly detection of web-based attacks, in: Proceedings of the 10th ACM conference on Computer and communications security, ACM, 2003, pp. 251–261.

[28] J. J. Davis, A. J. Clark, Data preprocessing for anomaly based network intrusion detection: A review, Computers & Security 30 (6) (2011) 353–375.

[29] M. Cova, C. Kruegel, G. Vigna, Detection and analysis of drive-by-download attacks and malicious javascript code, in: Proceedings of the 19th international conference on World wide web, ACM, 2010, pp. 281–290.

[30] K. Thomas, C. Grier, J. Ma, V. Paxson, D. Song, Design and evaluation of a real-time url spam filtering service, in: Security and Privacy (SP), 2011 IEEE Symposium on, IEEE, 2011, pp. 447–462.

[31] G. Yuan, B. Li, Y. Yao, S. Zhang, A deep learning enabled subspace spectral ensemble clustering approach for web anomaly detection, in: Neural Networks (IJCNN), 2017 International Joint Conference on, IEEE, 2017, pp. 3896–3903.

[32] S. Zanero, S. M. Savaresi, Unsupervised learning techniques for an intrusion detection system, in: Proceedings of the 2004 ACM symposium on Applied computing, ACM, 2004, pp. 412–419.

[33] P. Gogoi, B. Borah, D. K. Bhattacharyya, Anomaly detection analysis of intrusion data using supervised & unsupervised approach, Journal of Convergence Information Technology 5 (1) (2010) 95–110.

[34] B. Liu, Y. Xiao, L. Cao, Z. Hao, F. Deng, Svdd-based outlier detection on uncertain data, Knowledge and information systems (2013) 1–22.

[35] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural computation 13 (7) (2001) 1443–1471.

[36] J. Goldberger, Gaussian mixture model, URL: http://www. wisdom. weizmann. ac. il/~ irenak/ ncc 03/gmm. pdf.

[37] H. Liu, T. Liu, J. Wu, D. Tao, Y. Fu, Spectral ensemble clustering, in: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, 2015, pp. 715–724.

[38] L. Breiman, Random forests, Machine learning 45 (1) (2001) 5–32.

[39] W. Robertson, G. Vigna, C. Kruegel, R. A. Kemmerer, et al., Using generalization and characterization techniques in the anomaly-based detection of web attacks, in: NDSS, 2006.

[40] D. Yao, M. Yin, J. Luo, S. Zhang, Network anomaly detection using random forests and entropy of traffic features, in: Multimedia Information Networking and Security (MINES), 2012 Fourth International Conference on IEEE, 2012, pp. 926–929.

Bo Li received the B.E. and M.S. degrees from the Dalian University of Technology, Dalian, China, in 2002 and 2005, respectively, and the Ph.D. degree from Beihang University, Beijing, China, in 2011. He has been an Associate Professor with the School of Computer Science and Engineering, Beihang University, since 2018. His current research interests include computer security and Big Data.

Incorporating URL Embedding into Ensemble Clustering to Detect Network Anomalies

Bo Li[a,*], Guiqin Yuan[a], Li Shen[b], Ruoyi Zhang[c], Yiyang Yao[d]

[a]State Key Laboratory of Software Development Environment, Beihang University, Beijing, China

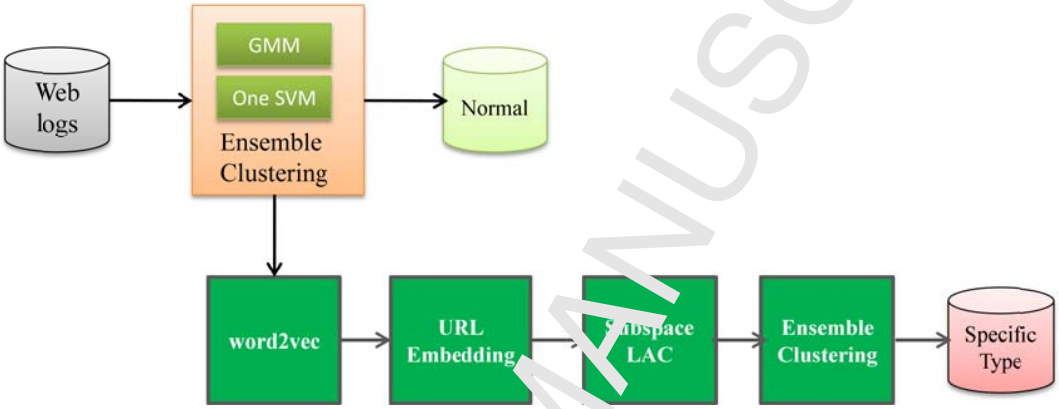
[b]Marketing Department, State Grid Hangzhou Power Supply Company, Hangzhou, China

[c]Regulation and Control Center, State Grid Hangzhou Power Supply Company, Hangzhou, China

[d]State Grid Zhejiang Electric Power Company Information Telecommunication Branch, Hangzhou, China

Correspondence: Bo Li, libo@act.buaa.edu.cn

Graphical Abstract



In Brief

We provide a weighted deep learning enabled subspace spectral ensemble clustering approach for web anomaly detection called WDL-SSEC. Our approach achieves better performance than existing approaches in terms of accuracy and performance and the experimental results demonstrate that our model has the ability to cluster anomalies into appropriate types.

Highlights

- We incorporate the word2vec into the multi-clustering model, which can extract semantic information of URLs and needn't specialized knowledge.
- We get efficient URL embedding by word embedding, which can solve the problem lacking professional stopping words in web anomaly detection.
- We extend the Spectral Ensemble Clustering (SEC) to get Subspace Weighted Ensemble Clustering (SWEC) by generating more quality basic partitioning.