Network Anomaly Detection Using Unsupervised Feature Selection and Density Peak Clustering

Xiejun Ni[1], Daojing He[1(✉)], Sammy Chan[2], and Farooq Ahmad[3]

[1] School of Computer Science and Software Engineering,
East China Normal University, Shanghai, China
djhe@sei.ecnu.edu.cn
[2] Department of Electronic Engineering,
City University of Hong Kong, Hong Kong, China
[3] Department of Computer Science,
COMSATS Institute of Information Technology, Lahore, Pakistan

Abstract. Intrusion detection systems (IDSs) play a significant role to effectively defend our crucial computer systems or networks against attackers on the Internet. Anomaly detection is an effective way to detect intrusion, which can discover patterns that do not conform to expected behavior. The mainstream approaches of ADS (anomaly detection system) are using data mining technology to automatically extract normal pattern and abnormal ones from a large set of network data and distinguish them from each other. However, supervised or semi-supervised approaches in data mining rely on data label information. This is not practical when the network data is large-scale. In this paper, we propose a two-stage approach, unsupervised feature selection and density peak clustering to tackle label lacking situations. First, the density-peak based clustering approach is introduced for network anomaly detection, which considers both distance and density nature of data. Second, to achieve better performance of clustering process, we use maximal information coefficient and feature clustering to remove redundant and irrelevant features. Experimental results show that our method can get rid of useless features of high-dimensional data and achieves high detection accuracy and efficiency in the meanwhile.

Keywords: Anomaly detection · Data mining · Feature selection · Maximal information coefficient · Density peak clustering

1 Introduction

Intrusion is a set of actions aiming to compromise the security of computer and network components in terms of confidentiality, integrity and availability [1]. Intrusion detection techniques can be classified into two categories:misuse detection (or signature-based detection) and anomaly detection. Misuse detection

M. Manulis et al. (Eds.): ACNS 2016, LNCS 9696, pp. 212–227, 2016.
DOI: 10.1007/978-3-319-39555-5_12

identifies intrusions based on patterns acquired from known attacks [2]. Anomaly detection discovers intrusions based on significant deviations from normal activities [3].

In early days, signature-based methods such as Snort [4], based on extensive knowledge of the particular characteristics of each attack, referred to as its signature are commonly applied. Such systems are highly effective in dealing with attacks for which they are programmed to defend unknown intrusion. Besides, they are not applicable for anomaly detection with large-scale network data because of the famous 4V [5]:

Volume. The scale and complexity of network data is beyond the Moores law which means the amount of traffic to be detected in every terminal increases rapidly. String matching based signature method is a computationally intensive task.

Variety. Network data usually is derived from various sources, where it is described in unstructured or semi-structured way. Proper integration is necessary to make uniform format.

Value. The value density of data is low. Anomaly detection problem usually faces with high dimensional network data. Some features of these data are useless in identifying anomaly.

Velocity. The detection needs response in real-time in order to detect attack or anomaly in time.

In addition, building new signatures require human experts' manual inspection which is not only expensive, but also induces a significant period of vulnerability between the discovery of a new attack and the construction of its signatures.

Patcha *et al.* [6] further categorizes anomaly detection methods into three categories: statistics-based, data mining-based and machine learning-based. Statistics-based method is difficult to adapt to the non-stationary variation of the network traffic, which leads to a high false positive rate [7]. To alleviates these shortcomings, a number of ADSs employ data mining techniques [8–12]. Data mining techniques aim to discover understandable patterns or models from given data sets [13]. It can efficiently identify profiles of normal network activities for anomaly detection, and build classifiers to detect attacks. Some earlier work show that these techniques can help to identify abnormal network activities efficiently.

Supervised anomaly intrusion detection approaches [8–10] highly rely on training data from normal activities, which are commonly used as data mining techniques. Since training data only contain historical activities, the profile of normal activities can only include the historical patterns of normal behavior. Therefore, new activities due to the change in the network environment or services are considered as deviations from the previously built profile, namely attacks. In addition, attack-free training data are not easy to obtain in real-world networks. The ADS trained by the data with hidden intrusions usually lacks the ability to detect intrusions.

To overcome the limitations of supervised anomaly-based systems, ADS employing unsupervised approaches has become a focus recently [14–17]. Unsupervised anomaly detection does not need attack-free training data. In distance-based methods, clusters are groups of data characterized by a small distance to the cluster center. However, a data point is always assigned to the nearest center, these approaches are not able to detect nonspherical clusters. In density-based spatial clustering methods, one chooses a density threshold, discards as noise the points in regions with densities lower than this threshold, and assigns to different clusters disconnected regions of high density. However, it can be nontrivial to choose an appropriate threshold.

Another challenge in ADS is feature selection. Many existing algorithms suffer from low effectiveness and low efficiency due to high dimensionality and large size of the data set. Hence, feature selection is essential for improving detection rate, since it can not only help reduce the computational cost but also improve the precision by removing irrelevant, mistaking and redundant features. However, in amount of data mining methods, features are selected based on the mutual information between feature and labels. Moreover, in many cases network data contain continuous variables which is challenging to measure the relation between features because the result greatly relies on the discretization methods.

Such limitations impose a serious bottleneck to unsupervised network anomaly detection problem. In this paper, we investigate anomaly detection problem in large scale and high-dimensional network data without labels and propose a new approach, called UFSDP (Unsupervised Feature Selection based Density Peak clustering) to tackle it. The major contributions of this paper are summarized as follows.

(1) We propose a new systematic framework that employs the density peak based clustering algorithm for network anomaly detection. This clustering algorithm has the advantage of extracting cluster centers and outlier points automatically. Besides, sampling adaptation is applied to improve the time and memory efficiency of the original clustering method in center selection stage.
(2) An unsupervised cluster-based feature selection mechanism is proposed before clustering procedure. We use two different ways to compute the relations for discrete and continuous attributes respectively. Different from other feature selection mechanism, we cluster the relevant features into groups according to their maximum redundancy from each other. Eventually redundant features are removed to make the feature number as least as possible.
(3) Extensive experiments are made to evaluate the performance of proposed method. Firstly, comparison are made over different classifiers by using original dataset and dataset with feature reduced by proposed selection algorithm. The proposed sampled-density peak clustering methodology is also compared with other clustering algorithms to evaluate its clustering performance in different credible metrics.

The rest of the paper proceeds as follows. Section 2 reviews related work. Section 3 describes our methodologies including unsupervised feature selection

and density peak clustering respectively and highlights our motivation in using them. Section 4 presents our evaluation results and analysis. Section 5 finally summarizes our work.

2 Related Work

2.1 Unsupervised Anomaly Detection

Most of current network anomaly detection systems are supervised learning method. However, training data is typically expensive to obtain. Using unsupervised anomaly detection techniques, the system can be trained with unlabeled data and is capable of detecting previously unseen attacks.

Clustering, a ubiquitous unsupervised learning method, aims to group objects into meaningful subclasses. Therefore, network data generated from different attack mechanism or normal activities have distinct characteristics so each of them can be distinguished from others.

KMeans, a clustering method, is employed to detect unknown attacks and divide network data space effectively in [17]. However the performance and computation complexity of KMean method are sensitive to the predefined number of clusters and initialized cluster centers. Wei *et al.* [18] employs improved FCM algorithms to obtain an optimal k.

In [19], the authors proposed an anomaly detection method. This method utilizes a density-based clustering algorithm DBSCAN for modeling the normal activities of a user in a host.

Egilmez *et al.* [16] proposed a novel spectral anomaly detection method by developing a graph-based framework over wireless sensor networks. In their method, graphs are chosen to capture useful proximity information of measured data and employed to project the graph signals into normal and anomaly subspaces.

In [20], a SOM-based anomaly intrusion detection system was proposed, which could contract high-dimension data to lower dimension, meanwhile keeping the primary relationship between clustering and topology. But results is sensitive to parameters such as neuron number.

2.2 Feature Selection

The machine learning community has developed many solutions to address the curse of dimensionality problem in the form of feature selection and feature extraction. Different from feature extraction methods such as principal component analysis (PCA) [21] and linear discriminant analysis (LDA) [22], feature selection methods aim to choose a representative subset of all the features instead of creating a subset of new features by combinations of the existing features, which reserves the interpretability of attributes.

Feature selection can be briefly divided into three broad categories: the filter, embedded and wrapper approaches. In terms of feature selection, filter methods are commonly used.

Filter algorithms have low computational complexity, but the accuracy of the learning algorithms is not guaranteed. In [23], Peng *et al.* propose a minimal-redundancy-maximal-relevance (mRMR) criterion, which adds a feature to the final subset if it maximizes the difference between its mutual information with the class and the sum of its mutual information with each of the individual features already selected. Qu *et al.* [24] suggested a new redundancy measure and a feature subset merit measure based on mutual information concepts to quantify the relevance and redundancy among features. Song *et al.* [25] proposed a feature filter FAST based on the mutual information between features and minimum spanning tree is used to split features into clusters. Only one representative feature will be selected from every cluster to form the best discriminative feature subset. But when all weight value of edges is not high enough to arise split, it is not applicable.

In addition, it lacks an effective way to compute the mutual information between continuous features. Since continuous variables have unlimited values and the probability of any of them is not defined. Equal-width [26] divides continuous value into a number of bins with equal width, however it can be inaccurate since the width is an uncertainty. Others uses parzen window [27] to estimate the probability density distribution of two variables and employ integration computation. The actual distribution is unknown and the result highly relies on the selection of kernel function. FSFC [28] applies a new similarity measure, called maximal information compression index as the measurement of feature similarity and also predefines the number of selected features in the final feature subset.

3 Methodology

3.1 Feature Selection

Feature selection is a commonly used technique to select relevant features by reducing the data dimensionality and building effective prediction models. Feature selection can improve the performance of prediction models by alleviating the effect of the curse of dimensionality, enhancing the generalization performance, speeding up the learning process.

Relevance Definition. Suppose F denotes the set of whole features, F_i denotes an element of F, C denotes the target concept and S_i denotes the F-F_i. There are mainly three kinds of features:

Definition 1 (Strong correlation). F_i is strong relevant to target concept C if and only if

$$p(C|S_i, F_i) \neq p(C|S_i) \tag{1}$$

Strong relevant features can have impact on distribution of classification. Lacking strong relevant features, the result would be inaccurate.

Definition 2 (Weak correlation). F_i is weak relevant to target concept C if and only if

$$p(C|S_i, F_i) = p(C|S_i),\ \exists S'_i \subset S_i, p(C|S'_i, F_i) \neq p(C|S'_i) \tag{2}$$

A weak relevant feature impacts the distribution of classification in some condition, but not necessary.

Definition 3 (Independent correlation). F_i is an independent feature if and only if

$$\forall S'_i \subset S_i, p(C|S'_i, F_i) \neq p(C|S'_i) \tag{3}$$

Independent features do not influence the distribution of classification, so they are firstly removed in feature selection.

Mutual Information Calculation. In previous work [23,25], the symmetric uncertainty is used as the measure of correlation between two features. The symmetric uncertainty is defined as follows:

$$SU(F_i, F_j) = \frac{2 * Gain(F_i, F_j)}{H(F_i) + H(F_j)} \tag{4}$$

$H(F_i)$ is the entropy of a discrete random variable $H(F_i)$, if $p(f)$ is the prior probabilities for all values of F_i, $H(F_i)$ is defined by:

$$H(F_i) = -\sum_{f \in F_i} p(f) \log_2 p(f) \tag{5}$$

$H(F_i, F_j)$ is the conditional entropy of F_i with priori knowledge of all values of F_j. The smaller $H(F_i, F_j)$ is, the greater $Gain(F_i, F_j)$ is:

$$Gain(F_i, F_j) = H(F_i) - H(F_i|F_j) = H(F_j) - H(F_j|F_i) \tag{6}$$

$Gain(F_i, F_j)$ means the contribution made by a known variable to reduce the uncertainty of an unknown variable, which can referred to another feature or the target concept.

Definition 4 (Relevancy). In supervised learning methods, features with low value of $SU(F_i, C)$ are firstly removed as independent ones. However, in unsupervised learning cases, the distribution of C are inaccessible. To deal with this problem, another measurement called ref is introduced to replace $SU(F_i, C)$ and their definition are as follows:

$$ref(F_i, C) = \frac{1}{n} \sum_{j=1}^{n} SU(F_i, F_j) \tag{7}$$

$$ref(F_i, F_j) = SU(F_i, F_j) \tag{8}$$

Discrete attributes such as *protocol_type* can directly be applied with aforementioned formulas. But continuous attributes such as *src_bytes* are uneasy to directly do so since their possible values are approximately infinite, and resulting in value $H(F_i)$ greater and value $SU(F_i, F_j)$ less than discrete attributes. As a result, it's challenging to compute relations between continuous features. Usually discretization operation is applied to map infinite values into finite values. However, most unsupervised discretization methods such as clustering and equal-width compute the relation in a rough way.

In this paper, the relation information between two continuous features are calculated using Maximal Information Coefficient (MIC) [29]. Methods such as mutual information estimators show a strong preference for some types of relations, but fails to describe well in other cases, which makes it unsuitable for identifying all potentially interesting relationships in a dataset. However, MIC has the ability to examine all potentially interesting relationships in a dataset independent of their form, which allows tremendous versatility in the search for meaningful insights.

MIC is based on the idea that if a relationship exists between two variables, then a grid can be drawn on the scatterplot of the two variables that divides the data to encapsulate that relationship. Given a finite dataset D of two dimensions, one of the dimensions named x-values and the other as y-values. Suppose x-values is divided into x bins and y-values into y bins, and we got a $x*y$ grid G, given by

$$I*(D,x,y) = argmaxI\,(D|G) \tag{9}$$

For each pair (x,y), the MIC algorithm finds the x by y grid with the highest induced mutual information. Then MIC algorithm normalizes the mutual information scores and compiles a matrix that stores $D|_G$. Then, the MIC(x,y) is the maximum value in the matrix.

Feature Cluster. After computing MI and MIC we get $ref(F_i, C)$ and $ref(F_i, F_j)$ from previous steps, then an intuitive clustering algorithm is proposed to filter those features. Firstly, features with low $ref(F_i, C)$ are removed since those features do not make obvious contribution for identifying. We set a $threshold1$ for $ref(F_i, C)$. In this paper, we run algorithm multiple times and choose the best one. After that, redundant features are removed according to the value of $ref(F_i, F_j)$. We set $threshold2$ for $ref(F_i, F_j)$, if $ref(F_i, F_j)$ exceeds $threshold2$, F_i and F_j can be regarded as redundant. Then we cluster those redundant features together. The details of the unsupervised feature selection algorithm for continuous features are given in Algorithm 1.

3.2 Density Peak Based Clustering

In distance-based methods, clusters are groups of data characterized by a small distance to the cluster center. However, a data point is always assigned to the nearest center, these approaches are not able to detect nonspherical clusters. In density-based spatial clustering methods, one chooses a density threshold,

Algorithm 1. Unsupervised continuous feature selection by MIC

Require: D=$\{F_0, F_1 ... F_{40}\}$ - the given dataset without label
$\theta 1$ - threshold for irrelevance
$\theta 2$ - threshold for redundancy
Ensure: S - selected feature subset

```
n = F_continuous.size()
M[n][n] = {0} //initialize the relevance matrix M
for each pair feature {F_i, F_j} do
    M[i][j] = M[j][i] = MIC[F_i][F_i]
end for
F_relevant = ∅
for i = 0 to n do
    M[i][i] = M[i][i] = Avg(M[i]) // M[i][i] is the relevance score of feature Fi, equal
    to the average value of M[i][0].. M[i][1]...M[i][n-1]
    if M[i][i] > θ1 then
        F_relevant = F_relevant ∪ F_i
    end if
end for
//=====Part1:Irrelevant Feature Removal=====
Feature_cluster = {}      //a  map
for for each F_i in F_relevant do
    if Feature_cluster = {} then
        Feature_cluster = Feature_cluster ∪ {i}
    else
        float maxredundancy = 0.0, int maxindex = 0
        for each Fj inFeature_cluster do
            if MIC[F_i][F_j] > maxreduncy then
                maxredundancy = MIC[F_i][F_j]
                maxindex = F_j.index
            end if
        end for
        if maxredundancy < θ2 then
            Feature_cluster = Feature_cluster[i] ∪ {i}
        else
            Feature_cluster[maxindex].insert(i)
        end if
    end if
end for
//=====Part2: Feature Clusters Construction=====
S = ∅
for each subset S' in Feature_cluster do
    F_j = max_{F_k ∈ S'} M[k][k]
    S = S ∪ F_j
end for
//=====Part3: Feature Selection=====
return S
```

discards as noise the points in regions with densities lower than this threshold, and assigns to different clusters disconnected regions of high density. However, it can be nontrivial to choose an appropriate threshold.

Most clustering algorithms [14–17] need parameters predefined, such as cluster number, and the detection accuracy is sensitive to those parameters. In [30], Alex *et al.* develop a modern clustering method named Fast Search and Find of Density Peaks (DP). Given data samples, there are two variables that does this algorithm calculates for each data sample.

(1) local density ρ_i:
ρ_i measures the local density of a target point i by computing the number of points within the fixed radius to point i. There are two ways to compute local density.
In cut-off kernel,

$$\rho_i = \sum_{j \in I_S \setminus \{i\}} \chi(d_{ij} - d_c) \tag{10}$$

$$\chi(x) = \begin{cases} 1, x < 0; \\ 0, x \geq 0, \end{cases} \tag{11}$$

In Gaussian kernel,

$$\rho_i = \sum_{j \in I_S \setminus \{i\}} e^{-\left(\frac{d_{ij}}{d_c}\right)^2} \tag{12}$$

(2) minimum distance to high density point δ_i:
δ_i is measured by computing the minimum distance between point i and any other point with higher density. The points with higher value of local density and distance are selected as cluster center.

Cluster Center Selection. In original density peak clustering, the density and distance of all the data samples are computed primarily. During this procedure, the method maintains a matrix with float number for distance in size of N*N where N is the number of samples. When N is higher than 32000, the memory can not store the whole matrix at one pass. Memory constraints density peak clustering to applied in a larger scale dataset. We notice that if we downsample the network data randomly, the whole distribution of data become sparse but the position of cluster centers remains changed slightly. Because the original data points with high density are still higher than other points after unbiased downsampling. Given this, we use a portion of network data instead of whole dataset and obtain approximate centers.

Clustering Process. After the cluster centers have been found, every remaining point is assigned to the nearest center. The label assignment is performed in a single step.

4 Experiments and Analysis

4.1 Dataset and Preprocess

KDDCup99 dataset [31] is used as a benchmark which contains five million connection records processed from four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. Due to the huge volume of original dataset, we use 10 % containing about 494021 records of this KDDCup99 dataset which is publicly available for experimental purpose. Attacks are broadly categorized in four groups such as Probes (information gathering attacks), DoS (denial of service), U2R (user to root) and R2L (remote to local). Each labeled record consists of 41 attributes (features) as depicted in Table 1 and one target value. Target value indicates the attack category name.

Algorithm 2. Data clustering by sampled Density-Peak algorithm

Require: D=$\{F_0, F_1 ... F_n\}$ - the dimension reduced dataset without label
m - sample reduce factor $Percent$ - position of d_c
$\theta 1$ - threshold for density $\theta 2$ - threshold for distance
Ensure: label - labels of data
for $i = 0$ *to* N **do**
if $random.(0, m) == 0$ **then**
$Sample.insert(D[i])$
end if
end for
=====Part1:Choose samples for centers =====
$List\ LL$
for *each pair* $(Sample[i], Sample[j])$ *in Samples* **do**
$dist[i][j] = eculidean\_distance(Sample[i], Sample[j])$
$LL.append(dist[i][j])$
end for
$d_c = percent * sorted(LL)$
for *each* i *in Sample* **do**
$Rho[i] = count_{j \in Sample\ \cap\ dist[i][j] < dc}(j)$
$Delta[i] = min_{j \in Sample\ \cap\ Rho[j] > Rho[i]}(dist[i][j])$
end for
for *each* i *in Sample* **do**
if $Rho[i] > \theta 1 \cap Delta[i] > \theta 2$ **then**
$Center.insert(i)$
end if
end for
=====Part2:Cluster center selection=====
$Label = [N]$
for *each* i *in* D **do**
$Label[i] = min_{j \in Centers}(eculidean\_distance(D[i], Center[j]))$
end for
=====Part3:Labeling=====
return $Label$

Table 1. Summay of the 41 attributes in KDDCup99 data sets

No	Feature name	Type	No	Feature name	Type
1	duration	C	22	is_guest_login	D
2	protocol_type	D	23	count	C
3	service	D	24	src_count	C
4	flag	D	25	serror_rate	C
5	src_bytes	C	26	srv_serror_rate	C
6	dst_bytes	C	27	rerror_rate	C
7	land	D	28	srv_rerror_rate	C
8	wrong_fragment	C	29	same_srv_rate	C
9	urgent	C	30	diff_srv_rate	C
10	hot	C	31	srv_diff_host_rate	C
11	num_failed_logins	C	32	dst_host_count	C
12	logged_in	D	33	dst_host_srv_count	C
13	num_compromised	C	34	dst_host_same_srv_rate	C
14	root_shell	D	35	dst_host_diff_srv_rate	C
15	su_attempted	D	36	dst_host_same_src_port_rate	C
16	num_root	C	37	dst_host_srv_diff_host_rate	C
17	num_file_creations	C	38	dst_host_serror_rate	C
18	num_shells	C	39	dst_host_srv_serror_rate	C
19	num_access_files	C	40	dst_host_rerror_rate	C
20	num_outbound_cmds	C	41	dst_host_srv_rerror_rate	C
21	is_hot_login	D			

Table 2. Specific of KDDCup99_10_percent

Attack category	Specific classes	No. of records
Normal	normal	97278
DoS	back,land,neptune,pod,smurf,teardrop	391458
Probe	ipsweep,nmapportsweep,satan	4107
R2L	ftpwriteguesspasswd,imap,multihop,phf,spy,warezclient...	1126
U2R	bufferoverflow,loadmodule,perl,rootkit	52
Total		494021

Since attributes in the KDD datasets include forms of continuous, discrete and symbolic with significantly varying resolution and ranges. In feature selection step, entropy and mutual information between discrete and symbolic attributes are computed without preprocessing. While in clustering stage, symbolic and discrete data are normalized and scaled. Firstly symbolic features like *protocol_type*, *services*, *flags* and *attack_names* were mapped to integer values ranging from

0 to $N-1$ where N is the number of symbols. Secondly, min-max normalization process is implemented. Each of feature is linearly scaled to the range of [0.0,1.0] for the fairness between different attributes. As we see in Table 2, the 10 % of KDDCup99 is an imbalanced dataset, with 'neptune', 'normal' and 'smurf' greatly higher than other kinds. Therefore we downsample three kinds to ensure the relative balance with other attributes.

4.2 Performance Evaluation

To evaluate the effectiveness and performance of our proposed method, simulation experiments have been carried out. All experiments are executed on a computer with Intel I5 CPU, CPU clock rate of 3.20 GHz, 4 GB main memory. The algorithm proposed is implemented with Winpython-64bit using programming language Python 2.7.9. Several valuable utilities, MINE package [32] and Python open source machine learning library Scikit-learn, Numpy, SciPy, Matplotlib [33] are adopted during experiments.

In feature selection stage, we present the experimental results in terms of the classification accuracy and the the time gain from reduced data to original. Parameters of Alrogithm 1 are setup as following: D=$KDDCup99\_10\_percent$, $\theta 1$=0.2, $\theta 2$=0.5. After running Algorithm 1, we obtained selected discrete feature subset $\{2, 3, 4, 12\}$ and continuous feature subset $\{1, 8, 10, 23, 24, 25, 26, 27, 28, 29, 32, 33\}$, totally 16 features with 60.97 % reduction compared to original features numbers. Our experiment is set up as follows:

1. Comparison is carried out over our unsupervised method with other feature selection approaches, including supervised such as RFE, ExtraTreeClassifier.
2. Five classification algorithms are employed to classify data before and after feature selection. They are the tree-based DecisionTreeClassifier, ensemble learning method ExtraTreesClassifier, Random Forest Classifier algorithm and AdaboostClassifier and optimal margin-based Support Vector Machine, respectively.
3. We sampled those three categoreis to obtain a balanced dataset and the total number of samples is about 20000. Given that the result can be different every time, we run the comparision experiments 100 times on the same machine and then obtain average measured values.

Figure 1 records the classification accuracy of five classifier achieved on datasets reduced by four feature selection methods. From it we observed that

1. The original data without feature selection achieve the highest accuracy in most classifier situation since it reserves all information of the whole data.
2. Most feature selection methods can achieve a high accuracy and is close to original data. In most case, ensemble learning model, Random Forest and AdaBoost methods can achieve better detection accuracy compared with other model, such as Decision Tree, Support Vector Machine.

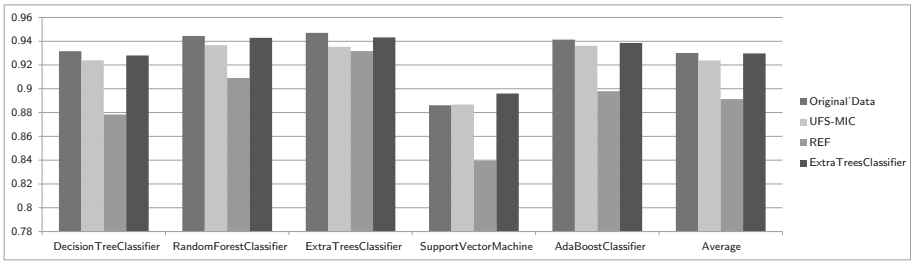

Fig. 1. Classification accuracy over different feature selection mehods

3. Compared with other supervised feature selection, MIC based-unsupervised feature selection acquire relatively high detection accuracy which is very close to the ExtraTreesClassifier with 0.4 % gap and to the original data with 0.6 % gap. Moreover, UFS-MIC achieves 3.3 % better than another supervised method RFE. The result shows that with absence of labels, the detection accuracy of proposed method is comparable with supervised approaches and thus suitable for network anomaly detection.

In the meanwhile, we record the time of running every classifier both features are selected and not. The detailed statistics in Table 3 illustrate that the proposed method efficiently reduces the time of running classification method on the reduced data. The average runtime benefit is considerable 14.44 % among different classifiers. In Decision Tree Classifier model, the benefit of 30.63 % is impressive.

Table 3. Runtime comparison between two datasets

	Orignal data	Reduced data	Time reduced
DecisionTreeClassifier	0.2520	0.1748	30.63 %
RandomForestClassifier	0.3969	0.3537	10.88 %
ExtraTreesClassifier	0.3782	0.3370	10.89 %
SupportVectorMachine	6.6171	5.8828	11.09 %
AdaBoostClassifier	22.4513	20.4912	8.73 %
Average	6.0191	5.4479	14.44 %

5 Conclusion

In this paper, we propose a two-stage framework for network anomaly detection. High-dimensional data commonly happens in network anomaly detection problems. Methods in solving these problem may suffer from curse of dimensionality.

In our first stage, we propose a sophisticated feature section method to get ride of irrelevant features and redundant features. By employing MIC approach, we solve the difficulty in calculating mutual information for continuous attributes. The experimental results show that this method achieves comparable accuracy with supervised methods and can effectively reduce the runtime of those methods with little sacrificing.

In the second stage, we introduce density peak based cluster. we have made a tradeoff that using fraction instead of the whole data samples to determine cluster centers approximatively. Experimental result shows that this method is efficient and achieve higher accuracy than other existing unsupervised methods generally.

Acknowledgement. This research is supported by the Pearl River Nova Program of Guangzhou (No. 2014J2200051), the National Science Foundation of China (Grants: 51477056 and 61321064), the Shanghai Rising-Star Program (No. 15QA1401700), the CCF-Tencent Open Research Fund, the Shanghai Knowledge Service Platform for Trustworthy Internet of Things (No. ZF1213), and the Specialized Research Fund for the Doctoral Program of Higher Education. Daojing He is the corresponding author of this article.

References

1. Heady, R., Luger, G.F., Maccabe, A., et al.: The architecture of a network level intrusion detection system. Department of Computer Science, College of Engineering, University of New Mexico (1990)
2. Barbara, D., Jajodia, S.: Applications of Data Mining in Computer Security. Springer Science & Business Media, New York (2002)
3. Eskin, E., Arnold, A., Prerau, M., et al.: A geometric framework for unsupervised anomaly detection. In: Barbará, D., Jajodia, S. (eds.) Applications of Data Mining in Computer Security, pp. 77–101. Springer, New York (2002)
4. Roesch, M.: Snort: lightweight intrusion detection for networks. LISA **99**(1), 229–238 (1999)
5. Camacho, J, Macia-Fernandez, G, Diaz-Verdejo, J., et al.: Tackling the big data 4 vs for anomaly detection. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 500–505. IEEE (2014)
6. Patcha, A., Park, J.M.: An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput. Netw. **51**(12), 3448–3470 (2007)
7. Luo, Y.B., Wang, B.S., Sun, Y.P., et al.: FL-LPVG: an approach for anomaly detection based on flow-level limited penetrable visibility graph (2013)
8. Tran, Q.A., Duan, H., Li, X.: One-class support vector machine for anomaly network traffic detection. China Education and Research Network (CERNET), Tsinghua University, Main Building, vol. 310 (2004)
9. Hu, W., Hu, W.: Network-based intrusion detection using Adaboost algorithm. In: The 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Proceedings, pp. 712–717. IEEE (2005)
10. Zhou, Q, Gu, L, Wang, C., et al.: Using an improved C4.5 for imbalanced dataset of intrusion. In: Proceedings of the 2006 International Conference on Privacy, Security, Trust: Bridge the Gap Between PST Technologies and Business Services, p. 67. ACM (2006)

11. Zhang, J., Zulkernine, M., Haque, A.: Random-forests-based network intrusion detection systems. IEEE Trans. Syst. Man Cybern Part C Appl. Rev. **38**(5), 649–659 (2008)
12. Tong, X., Wang, Z., Yu, H.: A research using hybrid RBF/Elman neural networks for intrusion detection system secure model. Comput. Phys. Commun. **180**(10), 1795–1801 (2009)
13. Hand, D.J., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press, Cambridge (2001)
14. Leung, K., Leckie, C.: Unsupervised anomaly detection in network intrusion detection using clusters. In: Proceedings of the Twenty-Eighth Australasian Conference on Computer Science, vol. 38, pp. 333–342. Australian Computer Society Inc (2005)
15. Zhang, J., Zulkernine, M.: Anomaly based network intrusion detection with unsupervised outlier detection. In: 2006 IEEE International Conference on Communications, ICC 2006, vol. 5, pp. 2388–2393. IEEE (2006)
16. Egilmez, H.E., Ortega, A.: Spectral anomaly detection using graph-based filtering for wireless sensor networks. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1085–1089. IEEE (2014)
17. Jianliang, M., Haikun, S., Ling B.: The application on intrusion detection based on k-means cluster algorithm. In: 2009 International Forum on Information Technology and Applications, IFITA 2009, vol. 1, pp. 150–152. IEEE (2009)
18. Jiang, W., Yao, M., Yan, J.: Intrusion detection based on improved fuzzy c-means algorithm. In: 2008 International Symposium on Information Science and Engineering, ISISE 2008, vol. 2, pp. 326–329. IEEE (2008)
19. Oh, S.H., Lee, W.S.: An anomaly intrusion detection method by clustering normal user behavior. Comput. Secur. **22**(7), 596–612 (2003)
20. Huang, S.Y., Huang, Y.N.: Network traffic anomaly detection based on growing hierarchical SOM. In: 2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 1–2. IEEE (2013)
21. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemometr. Intell. Lab. Syst. **2**(1), 37–52 (1987)
22. Yu, H., Yang, J.: A direct LDA algorithm for high-dimensional data with application to face recognition. Pattern Recogn. **34**, 2067–2070 (2001)
23. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. Pattern Anal. Mach. Intell. **27**(8), 1226–1238 (2005)
24. Qu, G., Hariri, S., Yousif, M.: A new dependency and correlation analysis for features. IEEE Trans. Knowl. Data Eng. **17**(9), 1199–1207 (2005)
25. Song, Q., Ni, J., Wang, G.: A fast clustering-based feature subset selection algorithm for high-dimensional data. IEEE Trans. Knowl. Data Eng. **25**(1), 1–14 (2013)
26. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Machine Learning: Proceedings of the Twelfth International Conference, vol. 12, pp. 194–202 (1995)
27. Kwak, N., Choi, C.H.: Input feature selection by mutual information based on Parzen window. IEEE Trans. Pattern Anal. Mach. Intell. **24**(12), 1667–1671 (2002)
28. Mitra, P., Murthy, C.A., Pal, S.K.: Unsupervised feature selection using feature similarity. IEEE Trans. Pattern Anal. Mach. Intell. **24**(3), 301–312 (2002)
29. Reshef, D.N., Reshef, Y.A., Finucane, H.K., et al.: Detecting novel associations in large data sets. Science **334**(6062), 1518–1524 (2011)

30. Rodriguez, A., Laio, A.: Clustering by fast search and find of density peaks. Science **344**(6191), 1492–1496 (2014)
31. Cup, K.: Data. knowledge discovery in databases darpa archive (1999)
32. Albanese, D., Filosi, M.: Mine tool. https://github.com/minepy/minepy
33. Pedregosa, F., Varoquaux, G., Gramfort, A., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)