Network Anomaly Detection Based on Semi-supervised Clustering

WEI XIAOTAO[1], HUANG HOUKUAN[2], TIAN SHENGFENG[2]
[1] School of Software, [2] School of Computer and Information Technology
Beijing Jiaotong University
Beijing 100044
CHINA

Abstract: A semi-supervised clustering algorithm based on the traditional k-means algorithm is proposed for network anomaly detection. We improve the original algorithm mainly in three aspects. First, the number of clusters is automatically decided by merging and splitting of clusters. Second, a small portion of labeled samples are employed to supervise the clustering process in the merging and splitting stage. Also, we modify the algorithm to directly process the symbolic attribute values. Experimental result on the KDD 99 intrusion detection datasets shows that our algorithm has high detection rate while maintaining a low false positive rate.

Key-Words: Network anomaly detection, Semi-supervised clustering, Grid-based clustering, K-means

1 Introduction

An intrusion detection system (IDS) is an automated system for the detection of computer system intrusions. There are two main classifications of IDSs. The first one divides the techniques of intrusion detection into two main types: anomaly and misuse detection. The anomaly detection approach establishes the profiles of normal activities of users, systems, system resources, network traffic and/or services and detects intrusions by identifying significant deviations from the normal behavior patterns observed from profiles. The misuse detection approach defines suspicious misuse signatures based on known system vulnerabilities and a security policy. According to the difference in monitoring objects, IDSs are divided into network-based IDSs and host-based IDSs. Host-based IDSs monitor a single host machine using the audit trails of a host operating system and network-based IDSs monitor any number of hosts on a network by scrutinizing the network traffic. In this paper, we focus on network anomaly detection (NAD).

The semi-supervised learning algorithm has recently attracted significant interest among researchers. In many fields such as data mining and network intrusion detection, there is typically a large amount of unlabeled data. In these applications labels can be extremely difficult or impossible to obtain (e.g. analysis of network traffic is very time-consuming and usually only a small portion of the available data can be labeled). However, in many cases, the information on labeled data is critical for the success of the clustering process and for the clustering accuracy. Consequently, learning with both labeled and unlabeled data has become the most suitable solution. Two sources of information are usually available to a semi-supervised clustering method: the domain background knowlage and a small portion of labeled data items or pairwise constraints. Most semi-supervised clustering algorithms are extensions of the well known k-means. Such as the constrained k-means algorithms [4, 5]. These algorithms rely on parameters that are difficult to set (such as the desired number of clusters) and require a high number of constraints to obtain significantly better results.

But for applying to network anomaly detection, a clustering algorithm must also be (1)efficient so that the run time of the algorithm scales well to the large number of traffic data records, (2)able to process data with mixed numeric and categorical values, (3)able to automatically determine the number of clusters.

In this paper we proposed a semi-supervised clustering algorithm to partition network traffic data. The first phase is a grid-based preclustering stage. The domain space is divided into un-overlapping d-dimensional cells. Only cells which actually contain data points are considered. This cell-based organization of the data will allow our algorithm to work efficiently on very large amounts of high-dimensional data. The second phase is a k-means like procedure. It can process data with mixed numeric and categorical values and automatically decided the cluster number by merging and splitting of clusters. The experimental result on KDD99 intrusion detection dataset proves the effectiveness of our algorithm.

The rest of this paper is organized as follows. In section 2, we describe our semi-supervised clustering algorithm. Section 3 introduces the KDD 99 IDS

datasets and the experimental result. Finally, some conclusions are given in Section 4.

2 The Semi-supervised Clustering Algorithm

Partition based clustering algorithms need to calculate the dissimilarity (commonly the Euclidean distance) of two vectors. However, traditional k-means can not deal with mixed symbolic attribute value and numerical attribute value. So we first define the distance measure on symbolic attributes and the related operations.

2.1 Processing Symbolic Attribute Values

In our algorithm, each symbolic attribute value is considered as a value set which contains only one member (the value of this attribute). We then define the summation of two symbolic values and the distance between two symbolic values as follows:

Def. 1. The summation of symbolic values is defined as the union set of these symbolic values.

Def. 2. The distance *d* between symbolic attribute value set A and B is defined as:

$$d = (1 - \frac{|A \cap B|}{|A \cup B|}) \cdot w \qquad (1)$$

w is the weight of this attribute. In this paper *w*=1 for all attributes for simplicity. Equation (1) can be used to calculate distances among clustering centers or distances between samples and clustering centers in nominal valued dimensions.

2.2 The Clustering Algorithm

Now we can propose our semi-supervised clustering algorithm.

Algorithm: Semi-supervised clustering algorithm for network anomaly detection.

Input:

The dataset $\{X_i, i = 1,2,...N\}$ which include a small portion of labeled data;

K = number of clusters desired;

I = maximum number of iterations allowed;

Output: Cluster centers and their radii.

Step 1. Arbitrarily choose *k* initial cluster centers: $M_1, M_2, ...M_k$ from the data set.

Step 2. Assign each of the *N* samples to the closest cluster center: $X \in \varpi_j$ if

$$D_j = \min(\|X - M_j\|, j = 1,...,k) .$$

While calculating the distance on symbolic attribute dimensions refer to Equation (1).

Step 3. (Supervised Splitting Stage) Scan labeled samples in all clusters. If there are any two labeled samples with opposite labels in one cluster, then split the cluster by adding two clusters with the two labeled samples being the cluster centers. Then delete the original cluster center. Let *k=k+1*. Go to Step 7.

Step 4. Update each cluster center:

$$M_j = \frac{1}{N_j} \sum_{X \in \varpi_j} X \quad (j=1,\ldots,k).$$

Step 5. Compute the pairwise distances D_{ij} between every two cluster centers:

$D_{ij} = \|M_i - M_j\|$, for all *i<>j*. and compute the θ_C: $\theta_C = \frac{1}{k!} \sum_{jl} D_{jl}$

Where *k! = k(k-1)(k-2)…1*, and

$D_{jl} = \|M_j - M_l\|$, *l = j+1, j+2, … ,k.*

Step 6. (Merging Stage) For all D_{ij}'s which are smaller than θ_C, perform pairwise merge: If there is no opposite labelled samples in M_i and M_j, Then merge them to form a new center: $M = \frac{1}{N_i + N_j}[N_i M_i + N_j M_j]$

Delete M_i and M_j, and let *k=k-1*.

Step 7. Terminate if maximum number of iterations I is reached or the number of clusters and their centers are not changed anymore. Otherwise go to Step 2.

After clustering we must label these clusters. For clusters that containing labeled samples, we can label the cluster with the sample's label. For clusters that containing no labeled samples, we randomly pick one sample out of the cluster and let the domain expert judge its label. Then we take this one sample's label as the cluster's label. We do not label a cluster by its size because the network traffic data do not fit the assumption that the majority of the network connections are normal traffic.

2.3 Intrusion Detection

In testing stage, if a testing sample is included in one cluster (the distance between the testing sample and the cluster center is less than the cluster's radius), then the label of the sample is determined by the label of this cluster. If a sample X is not included in any clusters, then we label this sample by the label of

cluster M_i where:

$$D(X, M_i) = \min_{i=1,2,...,k} \left(\frac{|X - M_i|}{r_i}\right) \quad (2)$$

Where r_i is the radius of the cluster M_i.

3 Experiment and Results

3.1 Description of KDD99 Intrusion Detection Datasets

The KDD 99 training dataset[12] contained about 5,000,000 connection records, and the training 10% dataset consisted of 494,021 records among which there were 97,278 normal connections (i.e. 19.69%). Each connection record consists of 41 different attributes that describe the different features of the corresponding connection, and the value of the connection is labeled either as an attack with one specific attack type, or as normal. There are 22 different attack types present in the 10% datasets. Each attack type falls exactly into one of the following four categories: Probing, DOS, U2R and R2L.

The task was to predict the value of each connection (normal or one of the above attack categories) for each of the connection record of the test dataset containing 311,029 connections. We use the training 10% dataset as training data and test the IDS with the 'Corrected' test dataset.

3.2 Data Preprocessing

Because the training 10% dataset contains 494,021 samples, we must firstly compress the dataset to keep the efficiency of the clustering algorithm. A good method to compress the training data is the grid-based method which divides the object space into finite number of cells called hyper-rectangles or units [1,9]. The main advantage of this approach is its fast processing time, which is typically dependent mainly on the number of cells in each dimension in the domain space.

In network connection records, there are numeric attributes and symbolic attributes. For symbolic attribute dimensions, such as 'protocol', we take each discrete value as a partition.

For numeric attributes, we characterize them into 3 types:

1. attributes only with binary values of 0 and 1
2. attributes that respect a percentage, or attributes with integer values and range in the hundreds
3. attributes with integer values and range in the thousands and above.

For the first type of attributes, we divide the dimensions into two partitions. For the second type of attributes, they are divided into N (in our approach N=10) equal length partitions. For the third type of attributes, equal length partition does not work. For example, in KDD 99 intrusion detection training 10% dataset, the values of "duration" attribute range from 0 to 58329. The average duration of normal records is 217.82 and the average duration of attack records is 6.34. When we divide this dimension into 10 equal length intervals, it seems that most records fall into the first interval, and this attribute has lost its ability for classification. Therefore we transform these attribute values onto a range of (0,1) with a sigmoid function as formula 3.

$$f(x) = \frac{1}{1 + e^{-(x-m)/c}} \quad (3)$$

where $m = \frac{1}{2}(m_n + m_a)$, $c = \frac{1}{2}(d_n + d_a + |m_n - m_a|)$. m_n and d_n denote the average value and standard deviation of normal records on this dimension, m_a and d_a denote the average value and standard deviation of attack records on this dimension. They can all be calculated from the history network traffic or obtained by domain background knowledge. And then the new value are split into N equal length intervals.

we process the training 10% data with the approach described above. The 494021 records are projected into 19461 populated cells. In order to check the effect of our method, we scan all the records' label information in each cell and find that among the 19461 cells, only 33 cells containing both normal records and attack records, which totally involving 2018 records (1304 normal records and 714 attack records) that account for 0.41% of the 494021 records. In fact, we do not need the label information. We only want to prove that the preprocessing indeed compressed the network traffic data with a little information loss. Also the granularity of the cell is fine enough for N=10.

3.3 Experimental Results

We input the 19461 samples into our clustering algorithm with 194 densest samples being labeled. We set the parameter I=100, and try the parameter K from 5 to 20. Every time the algorithm returns the result less than 17 iterations. And always we get 31 clusters even though the cluster centers and their radii are slightly deferent. Among them there are 12

clusters labeled as intrusion and the other 19 clusters labeled as normal.

We test the clustering result with the "Corrected" testing dataset according to Equation (2). The detection rate is 94.42% and the false positive rate is only 1.52%. To our knowledge, this is the best result compared to other clustering-based intrusion detection methods on the whole "Corrected" dataset. The result can even compete with some supervised learning methods. Table 1 provides a summary of some recent results from alternative approaches trained on the KDD 99 dataset and tested using the 'Corrected' dataset. It demonstrates that our semi-supervised clustering algorithm is well applied in network intrusion detection task.

Table 1. Recent result on the KDD benchmark

Technique	Detection Rate	FP Rate
Clustering	93%	10%
K-NN	91%	8%
Rough Set	93.5%	2.8%
SVM	98%	10%
SOM	96.1%	7.8%

4 Conclusion

In this paper, we proposed a semi-supervised clustering algorithm for network traffic data. Labeled samples are used to supervise the clustering process. Related operations for symbolic values are defined so that the algorithm can process mixed symbolic attribute values and numerical attribute values. The experimental result on KDD 99 intrusion detection datasets shows that our algorithm was able to accurately discover clusters on the training dataset. The intrusion detection testing on "Corrected" dataset shows that our algorithm has high detection rate and relatively low false positive rate.

References:

[1] Zhiwen Y, Hau-San W.: GCA: A Real-time grid-based clustering algorithm for large data set, Proceedings of the 18th International Conference on Pattern Recognition, 2006

[2] Tom C, DongPing F, John C, Yao W, CHristonpher J, A robust and scalable clustering algorithm for mixed type attributes in large database environment. Proc. of the 7th ACM SIGKDD international conference on knowledge discovery and data mining, 2001

[3] Jie Bai, Yu Wu, Guoyin Wang, Simon X. Yang, and Wenbin Qiu.: A Novel Intrusion Detection Model Based on Multi-layer Self-Organizing Maps and Principal Component Analysis, . Proceedings of The 3rd International symposium on Neural Networks, LNCS3973, 2006

[4] Basu, S., Banerjee, A., and Mooney, R.J.: Semi-supervised clustering by seeding. Proc. 19th Int. Conf. on Machine Learning, ICML'02, 2002

[5] K. Wagstaff, C. Cardi, S. Rogers, S. Schroedl, Constained K-means with Background Knowledge, in Proc. Eighteenth Int. Conf. on Machine Learning, 2001

[6] Kayacik, G. H., Zincir-Heywood, A. N., Heywood, M. I.: On the Capability of an SOM Based Intrusion Detection System. In: Proceedings of the 2003 IEEE IJCNN. Portland, USA (2003) 1808-1813

[7] Raymond T.Ng, Jiawei Han.: Efficient and Effective Clustering Methods for Spatial Data Mining. Department of Computer Science of University of British Columbia Vancouver,B.C.,V6T 124, Canada, School of Computing Sciences of Simon Fraser University Burnaby, B.C., V5A 1S6,Canada:144-155,1994.

[8] M. Halkidi and M. Vazirgiannis.: Clustering validity assessment:Finding the optima partitioning of a data set . In ICDM, 2001.

[9] Wang, W., Yang, J., Muntz, R. R.: Sting: A Statistical Information Grid Approach to Spatial Data Mining. In: Proceedings of the 23rd International Conference on Very Large Data Bases. Athens, Greece (1997) 186-195

[10] Yu G, Ali A. G, Nabil B. K-means+: An autonomous clustering algorithm. Technical Report TR04-164, University of New Brunswick, 2004.

[11] Zhexue H, Extensions to the k-means algorithm for clustering large data sets with categorical values, Data Mining and Knowledge Discovery, 1998,v2 pp283-304

[12] "KDD Cup 1999Data", Machine Learning for Intrusion Detection Project. The University of Columbia, Fall 2000, http://www.cs.columbia.edu/ids/ml/