A Unified Subspace Outlier Ensemble Framework for Outlier Detection

Zengyou He, Shengchun Deng, and Xiaofei Xu

Department of Computer Science and Engineering,
Harbin Institute of Technology, China
zengyouhe@yahoo.com, {dsc, xiaofei}@hit.edu.cn

Abstract. This paper proposes a unified framework for outlier detection in high dimensional spaces from an ensemble-learning viewpoint. Moreover, to demonstrate the usefulness of our framework, we developed a very simple and fast algorithm, namely SOE1, in which only subspaces with one dimension is used for mining outliers from large categorical datasets. Experimental results demonstrate the superiority of SOE1 algorithm.

1 Introduction

Most applications for outlier mining are high dimensional domains in which the data may contain hundreds of dimensions. In this paper, we propose a unified framework for outlier detection in high dimensional spaces from an ensemble-learning viewpoint.

In our new framework, the outlying-ness of each data object is measured by fusing outlier factors in different subspaces using a combination function. In addition, to demonstrate the usefulness of the ensemble-learning based outlier detection framework, we developed a very simple and fast algorithm, namely SOE1 (**S**ubspace **O**utlier **E**nsemble using **1**-dimensional Subspaces) in which only subspaces with one dimension is used for mining outliers from large categorical datasets. The SOE1 algorithm needs only two scans over the dataset and hence is very appealing in real data mining applications. Experimental results on real datasets show that SOE1 has comparable performance with respect to those state-of-art outlier detection algorithms on identifying true outliers.

2 Problem Formulation and Unified Framework

Let D be a database of d-dimensional feature vectors. An element $P \in D$ is called point or object. Let $A = \{A_1, A_2, \ldots, A_d\}$ be the set of all attributes A_i of D. Any subset $S \subseteq A$, is called a subspace. The cardinality of S ($|S|$) is called the dimensionality of S. The power set of A, denoted by Pow (A), is defined as Pow (A)= $\{S \mid S \subseteq A\}$. Hence, each subspace is an element of Pow (A). The projection of an object P into a subspace $S \in Pow$ (A) is denoted by $\pi_S(P)$. The outlier factor of an object P in subspace S is denoted by $OF(\pi_S(P))$.

W. Fan, Z. Wu, and J. Yang (Eds.): WAIM 2005, LNCS 3739, pp. 632 – 637, 2005.

The problem of outlier detection in high dimensional space and the unified ensemble learning based algorithmic framework are described in Fig.1.The input for outlier detection in high dimensional space includes the target database, the number of desired outliers, the set of subspaces considered in the mining process and the combination function. Among all these input parameters, the set of subspaces and combination function are of primary importance.

Outlier Detection in High Dimensional Space:

Mining top-k outliers from a database using a set of subspaces and a combination function

Input:

(1): A database D with set of features A

(2): An Integer k, i.e., the k most outlying objects to be mined

(3): SS, a set of subspaces, i.e., SS is a subset of $Pow(A)$

(4): A combination/ensemble function $\oplus$

Output:

Top-k outliers that satisfy the requirement

Unified Algorithmic Framework:

(1) Individual subspace outlier factor computation step

For each subspace S in SS

For each object P in D

Compute the outlier factor of P in S, i.e., $OF(\pi_S(P))$

(2) Outlier ensemble step

For each object P in D

Ensemble all the outlier factors of P in different subspaces, i.e., $OF(P) = \bigoplus_{S \in SS} OF(\pi_S(P))$

Fig. 1. The unified subspace outlier ensemble based algorithmic framework-SOE framework

The unified algorithmic framework (subspace outlier ensemble (SOE) framework) consists of two steps: subspace outlier mining and subspace outlier ensemble.

In the subspace outlier-mining step, the SOE framework uses existing outlier mining algorithms to compute the outlier factors of data objects in all the input subspaces.

In subspace outlier ensemble step, we borrow some ideas from ensemble learning by fusing outlier factors in different subspaces using a combination function. Hence, the choice of combination function (or combining operator) is at the core of the outlier ensemble stage.

Suppose the outlier factors of an object P in D in different subspaces are denoted as $v_1, v_2, \ldots, v_m$ (the number of input subspaces is m). And the combining operator is denoted as $\oplus$. By fusing all the subspace outlier factors, the final outlier factor of P is $OF(P) = \oplus(v_1, v_2, \ldots, v_m)$. Note that if $m=1$, $\oplus(v_1, v_2, \ldots, v_m) = v_1$.

Our potential choices for $\oplus$ are the followings (which are also used in [7] for class outlier mining).

- The product operator $\prod$: $\oplus(v_1, v_2, \ldots, v_m) = v_1 v_2 \ldots v_m$.
- The addition operator +: $\oplus(v_1, v_2, \ldots, v_m) = v_1 + v_2 + \ldots + v_m$.

- A generalization of addition operator-it is called the S_q combining rule, where q is an integer number. $S_q\ (v_1, v_2, \ldots, v_m) = \left(v_1^q + v_2^q + \ldots + v_m^q\right)^{(1/q)}$. Note that the addition is simply the S_1 rule.
- A "limiting" version of S_q rules, denoted as S_∞. $S_\infty(v_1, v_2, \ldots, v_m)$ is defined to be equal to v_i, where v_i has the largest absolute value among $(v_1, v_2, \ldots, v_m)$.

3 SOE1 Algorithm

Let D be a database of d-dimensional feature vectors. Let $A = \{A_1, A_2, \ldots, A_d\}$ be the set of all attributes A_i of D. The value set V_i is set of values of A_i that are present in D. For each attribute value $v \in V_i$, the frequency $f(v)$, denoted as f_v, is number of objects $P \in D$ with $P.A_i = v$. The number of distinct attribute values of A_i is supposed to be p_i. We define the histogram of A_i as the set of pairs: $h_i = \{(v_1, f_1), (v_2, f_2), \ldots, (v_{p_i}, f_{p_i})\}$. Each element of h_i is called an entry in the histogram or just a histogram entry. The histogram of the dataset D is defined as: $H = \{h_1, h_2, \ldots, h_d\}$.

The proposed SOE1 algorithm needs only two scans over the dataset. The first scan of SOE1 is the subspace outlier-mining step, in which we construct the histogram of the dataset D. Intuitively, in one-dimensional space the outlying-ness of an object is determined by the occurrences of its corresponding attribute value, i.e., higher frequency implies more normal the object is. Hence, the outlier factor of each object $P \in D$ in subspace A_i is the frequency $f(P.A_i)$. Hence, $S_\infty(v_1, v_2, \ldots, v_m)$ is modified to be equal to v_i, where v_i has the smallest absolute value among $(v_1, v_2, \ldots, v_m)$. To store the histogram of the dataset D, we need d hash tables as our basic data structures (each hash table for one histogram of A_i). Actually, each hash table is the materialization of a histogram. Therefore, we will use histogram and hash table interchangeably in the remaining parts of the paper.

The second scan of SOE1 is the subspace outlier-ensemble step, in which we aggregate the outlier factors in different one-dimensional subspaces using a combination function. That is, for each object $P \in D$, we retrieve its frequencies of attribute values, i.e., outlier factors, from hash tables efficiently. Then, fusing these outlier factors to get final outlying-ness. To report the top-k outliers, we maintain a k-length array for this purpose.

4 Experimental Results

We used three real life datasets from UCI [5] to demonstrate the effectiveness of our algorithm against *FindFPOF* algorithm [1], *FindCBLOF* algorithm [2] and *KNN* algorithm [3].

For all the experiments, the two parameters needed by *FindCBLOF* algorithm are set to 90% and 5 separately as done in [2]. For the *KNN* algorithm [3], the results were obtained using the *5-nearest-neighbour*; For *FindFPOF* algorithm [1], the parameter *mini-support* for mining frequent patterns is fixed to 10%, and the maximal number of items in an itemset is set to 5. Since the SOE1 algorithm is parameter-free,

we don't need to set any parameters. Furthermore, we empirically study the impact of different combining operators on SOE1. That is, in the experiments, we report the results of SOE1 with different combining operators. For S_q operator, we set q to 2, 5 and 7 separately.

Since we know the true class of each object in the test datasets, we define objects in small classes as rare cases (i.e., outliers). The number of rare cases identified is utilized as the assessment basis for comparing our algorithm with other algorithms.

The first dataset used is the Lymphography dataset, which has 148 instances with 18 attributes. The data set contains a total of 4 classes. Classes 2 and 3 have the largest number of instances. The remained classes are regarded as rare class labels for they are small in size. The corresponding class distribution is illustrated in Table 1.

Table 1. Class Distribution of Lymphography Dataset

Case	Class codes	Percentage of instances
Commonly Occurring Classes	2, 3	95.9%
Rare Classes	1, 4	4.1%

Table 2 shows the results produced by different algorithms. Here, the *top ratio* is ratio of the number of records specified as *top-k* outliers to that of the records in the dataset. For example, we let SOE1 (+) algorithm find the *top 16* outliers with the top ratio of 11%. By examining these 16 points, we found that 6 of them belonged to the rare classes.

Table 2. Detected Rare Classes in Lymphography Dataset

Top Ratio (Number of Records)	Number of Rare Classes Included								
	SOE1 ($\prod$)	SOE1 (+)	SOE1 (S_q)			SOE1 (S_∞)	*Find FPOF*	*Find CBLOF*	*KNN*
			q=2	q=5	q=7				
5% (7)	6	5	4	4	4	2	5	4	4
10%(15)	6	6	5	4	4	6	5	4	6
11%(16)	6	6	5	4	4	6	6	4	6
15%(22)	6	6	5	5	4	6	6	4	6
20%(30)	6	6	6	5	4	6	6	6	6

One important observation from Table 2 was that, among all the potential choices of $\oplus$ we are considered in SOE1, the + operator and $\prod$ operator are the clear winners in this experiment. That is, SOE1 with the + operator and $\prod$ operator outperform S_q and S_∞ in all cases. This observation suggests that the + operator and $\prod$ operator will be better choices in practice for users. Consequent experiments also support similar conclusions. Moreover, with increase of q in the S_q operator, the performance of SOE1 will deteriorate.

Furthermore, in this experiment, the SOE1 algorithm with $\prod$ operator performed the best for all cases and can find all the records in rare classes when the *top ratio*

reached 5%. In contrast, for the *FindFPOF* algorithm, it achieved this goal with the *top ratio* at 10%, which is almost the twice for that of our algorithm.

The second dataset used is the Wisconsin breast cancer data set, which has 699 instances with 9 attributes. Each record is labeled as *benign* (458 or 65.5%) or *malignant* (241 or 34.5%). We follow the experimental technique of Harkins, et al. [6] by removing some of the *malignant* records to form a very unbalanced distribution; the resultant dataset had 39 (8%) *malignant* records and 444 (92%) *benign* records (http://research.cmis.csiro.au/rohanb/outliers/breast-cancer/). The corresponding class distribution is illustrated in Table 3.

Table 3. Class Distribution of Wisconsin Breast Cancer Dataset

Case	**Class codes**	**Percentage of instances**
Commonly Occurring Classes	1	92%
Rare Classes	2	8%

Table 4. Detected Malignant Records in Wisconsin Breast Cancer Dataset

Top Ratio (Number of Records)	Number of Rare Classes Included									
	SOE1 with different operators						*Find FPOF*	*Find CBLOF*	*RNN*	*KNN*
	$\prod$	+	(S_q)			S_∞				
			q=2	q=5	q=7					
1%(4)	4	4	3	3	3	3	3	4	3	4
2%(8)	7	7	7	7	7	5	7	7	6	8
4%(16)	15	14	14	14	14	11	14	14	11	16
6%(24)	22	21	19	19	16	17	21	21	18	20
8%(32)	27	28	26	25	23	23	28	27	25	27
10%(40)	33	32	31	30	28	28	31	32	30	32
12%(48)	36	36	34	33	33	33	35	35	35	37
14%(56)	39	39	38	37	37	37	39	38	36	39
16%(64)	39	39	39	38	38	38	39	39	36	39
18%(72)	39	39	39	39	39	38	39	39	38	39
20%(80)	39	39	39	39	39	39	39	39	38	39
25%(100)	39	39	39	39	39	39	39	39	38	39
28%(112)	39	39	39	39	39	39	39	39	39	39

For this dataset, we also consider the *RNN* algorithm [6]. The results of *RNN* algorithm on this dataset are reproduced from [6]. Table 4 shows the results produced by the different algorithms. Clearly, SOE1 with + operator and $\prod$ operator also outperform SOE1 with S_q and S_∞ in all cases on this dataset. Furthermore, among all of these algorithms, *RNN* performed the worst in most cases. Compared to other algorithms, SOE1 (with + operator and $\prod$ operator) achieves roughly the same average performance with respect to the number of outliers identified.

Arrhythmia data is the third dataset used in our experiments, which has 279 attributes. The dataset contains a total of 13 non-empty classes. As suggested in [4],

class labels that occurred less than 5% of the dataset are considered as rare classes. The corresponding class distribution is illustrated in Table 5.

Since most attributes in this dataset are continuous, hence, we first perform a grid discretization of the data. Each attribute is divided into 2 equal-width bins.

We let each algorithm report top 85 outliers, as done in [4]. Among these reported data objects, we examine how many of them belong to rare classes. Table 6 shows the results produced by the different algorithms.

From Table 6, we can see that the algorithm in [4] produced the best result with the cost of much more running time. In the remaining algorithms, most SOE1 variations are slight better, at least achieved the same level performance. Although the performance of SOE1 algorithm on this dataset is not so good as that of the algorithm in [4], it is at least acceptable.

Table 5. Class Distribution of Arrhythmia Dataset

Case	**Class codes**	**Percentage of instances**
Commonly Occurring Classes (>=5%)	01,02,06,10,16	92%
Rare Classes (<5%)	03,04,05,07,08,09,14,15	8%

Table 6. Detected Rare Classes in Arrhythmia Dataset

Number of Records	Number of Rare Classes Included									
	SOE1 ($\prod$)	SOE1 (+)	SOE1(S_q)			SOE1 (S_∞)	*Find FPOF*	*Find CBLOF*	[4]	*KNN*
			q=2	q=5	q=7					
85	33	32	33	34	33	27	32	32	43	28

5 Conclusions

From an ensemble-learning viewpoint, a unified subspace outlier ensemble framework for outlier detection in high dimensional spaces is proposed in this paper. Empirical evidence verified the feasibility and advantage of our method.

References

1. He, Z., Xu, X., Huang, J., Deng, S.: A Frequent Pattern Discovery Based Method for Outlier Detection. In: Proc. of WAIM'04, pp. 726-732, 2004
2. He, Z., Xu, X., Deng, S.: Discovering Cluster Based Local Outliers. Pattern Recognition Letters, 2003, 24(9-10): 1641-1650
3. Ramaswamy, S., Rastogi, R., Kyuseok, S.: Efficient Algorithms for Mining Outliers from Large Data Sets. In: Proc. of SIGMOD'00, pp. 93-104, 2000
4. Aggarwal, C., Yu, P.: Outlier Detection for High Dimensional Data. In: Proc. of SIGMOD'01, pp. 37-46, 2001
5. Merz, G., Murphy, P.: Uci Repository of Machine Learning Databases. http://www.ics. uci.edu/ mlearn/MLRepository.html, 1996
6. Harkins, S., et al.: Outlier Detection Using Replicator Neural Networks. In: Proc. of DaWaK'02, pp. 170-180, 2002
7. He, Z., Xu, X., Huang, J., Deng, S.: Mining Class Outliers: Concepts, Algorithms and Applications in CRM. Expert System With Applications, 2004, 27(4): 681-697