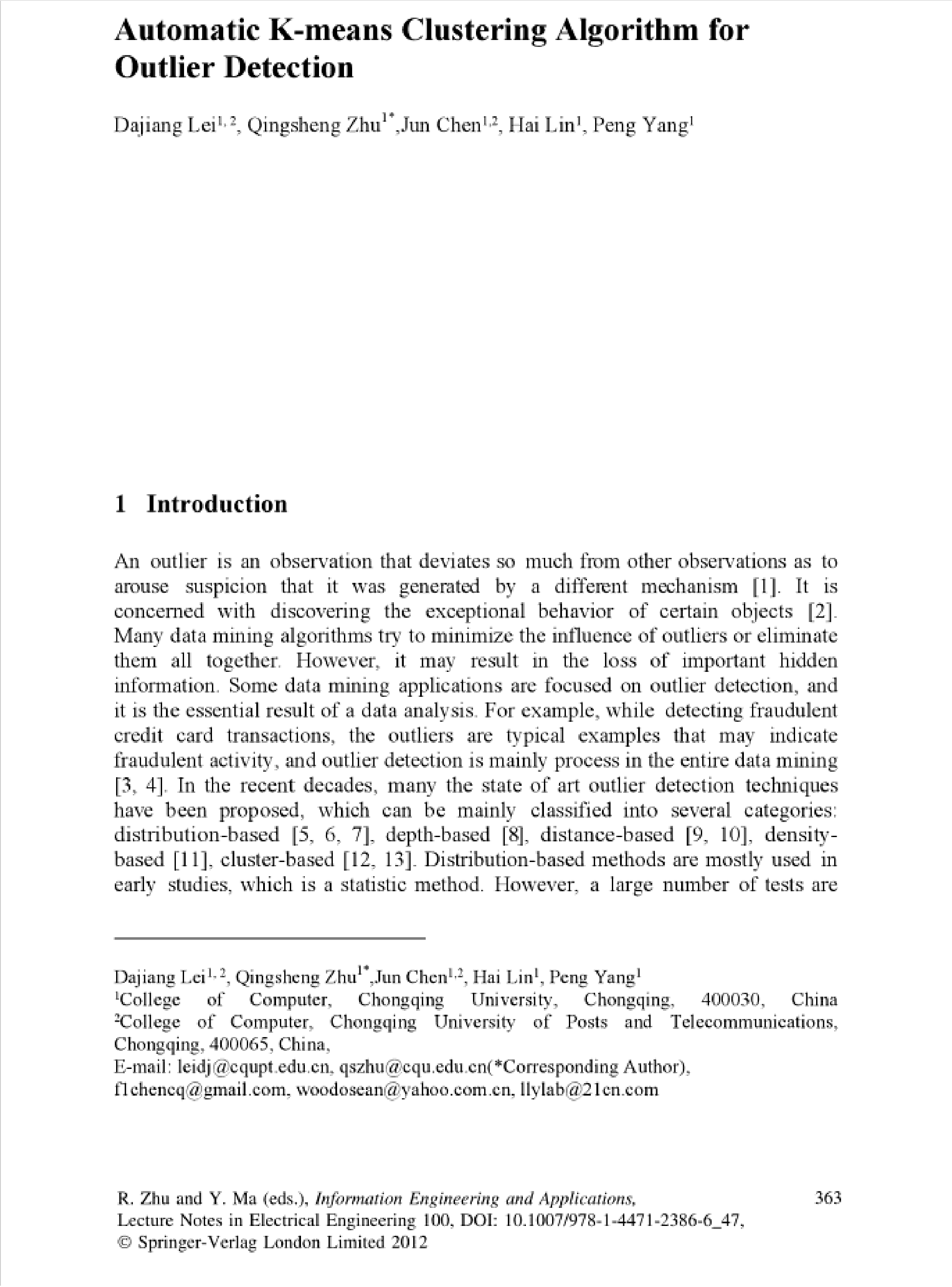
http://books.google.com.tw/books?hl=en&lr=&id=fKAzECEfd-0C&oi=fnd&pg=PA363&dq=outlier+detection&ots=e0GIdowo5M&sig=aqV4y9rzAcY5kHV-TKEMDm7BXxs&redir_esc=y#v=onepage&q&f=true



Automatic K-means Clustering Algorithm for Outlier Detection

Dajiang Lei[1, 2], Qingsheng Zhu[1*],Jun Chen[1,2], Hai Lin[1], Peng Yang[1]

1 Introduction

An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism [1]. It is concerned with discovering the exceptional behavior of certain objects [2]. Many data mining algorithms try to minimize the influence of outliers or eliminate them all together. However, it may result in the loss of important hidden information. Some data mining applications are focused on outlier detection, and it is the essential result of a data analysis. For example, while detecting fraudulent credit card transactions, the outliers are typical examples that may indicate fraudulent activity, and outlier detection is mainly process in the entire data mining [3, 4]. In the recent decades, many the state of art outlier detection techniques have been proposed, which can be mainly classified into several categories: distribution-based [5, 6, 7], depth-based [8], distance-based [9, 10], density-based [11], cluster-based [12, 13]. Distribution-based methods are mostly used in early studies, which is a statistic method. However, a large number of tests are

Dajiang Lei[1, 2], Qingsheng Zhu[1*],Jun Chen[1,2], Hai Lin[1], Peng Yang[1]
[1]College of Computer, Chongqing University, Chongqing, 400030, China
[2]College of Computer, Chongqing University of Posts and Telecommunications, Chongqing, 400065, China,
E-mail: leidj@cqupt.edu.cn, qszhu@cqu.edu.cn(*Corresponding Author), f1chencq@gmail.com, woodosean@yahoo.com.cn, llylab@21cn.com

R. Zhu and Y. Ma (eds.), *Information Engineering and Applications*,
Lecture Notes in Electrical Engineering 100, DOI: 10.1007/978-1-4471-2386-6_47,

often required to decide which distribution model fits the arbitrary dataset best. Fitting the data with standard distributions is costly, and may not produce satisfactory results. The second category of outlier studies in statistics is depth-based. Each data object is represented as a point in a *k-d* space, and is assigned a depth. With respect to outlier detection, outliers are more likely to be data objects with smaller depths. However, in practice, depth-based approaches become inefficient for large datasets for $k \geq 4$. Hence, many other categories of methods are proposed. In [9], Knorr and Ng firstly proposed the notion of distance-based outliers which is a non-local approach and can not find local outliers in complex structure datasets. Then, Breunig et al. propose density-based local outliers detection method (LOF) based on the distance of a point from its k nearest neighborhood and declare the top n points in this ranking to be outliers. Furthermore, many clustering algorithms, especially those developed in the context of KDD were extended to have capable of handling exceptions. The ordinary clustering based outlier detection methods find outliers as a side-product of clustering algorithm, which regard outliers as objects not located in clusters of dataset. Even though clustering and outlier detection appear to be fundamentally different from each other, there are numerous studies on clustering-based outlier detection methods. Many data-mining algorithms in literature detect outliers as a by-product of clustering algorithms themselves, which define outliers as points that do not lie in or located far apart from any clusters. Actually no clustering algorithm can precisely classify every data instance and some special data points in a certain cluster may be outliers. In this paper, we present an improved cluster based local outlier factor (***CBLOF***) [12] to tackle this problem. For clustering based outlier detection algorithms, the number of clusters is needed to choose. However, for unknown data sets, we choose the number of clusters arbitrarily and it would decrease the performance of the algorithm. In this paper, we combine k-means clustering algorithm with a cluster validation metric to propose an automatic k-means clustering algorithms for dividing data instances into *k* clusters. To our proposed algorithm, the accurate cluster structure (or approximate to the actual nature of data set) is crucial to improve detecting ratio and false alarm ration in the context of outlier detection.

The rest of this paper is organized as follows. Section 2 introduces subtractive clustering algorithm for estimating approximate number of clusters. Section 3 introduces the definition of the cluster validation metric used in our proposed algorithm and propose automatic k-means clustering algorithm. Section 4 proposes the improved ***CBLOF*** and outlier detection algorithm based automatic k-means clustering algorithm. Section 5 elaborates the experiments for demonstrating the effectiveness and efficiency of our proposed method. We conclude the paper in Section 6.

2 Subtractive Clustering

The Subtractive Clustering (***SC***) method is adopted in this paper to estimate approximate number of clusters of a given data set. Suppose that we don't have a clear idea how many clusters there should be for a given data set. Subtractive clustering [14] is a fast, one-pass algorithm for estimating the number of clusters and the cluster centers in a set of data. The subtractive clustering algorithm is described as follows. Consider a group of *n* data points $\{x_1,x_2,\ldots,x_N\}$, where x_i is a data point, denoted as a vector in the feature space. Without loss of generality, we assume that the feature space is normalized so that all data are bounded by a unit hypercube. Firstly, we consider each data point as a potential cluster center and define a measure of the point to serve as a cluster center. The potential of x_i, denoted as P_i, is computed as by Eq. 1.

$$P_i = \sum_{j=1}^{n} \exp(-\alpha \|x_i - x_j\|^2) \ . \qquad \textbf{(1)}$$

$$P_i = P_i - P(x_k^*) \exp\left(-\beta \|x_i - x_k^*\|^2\right) . \qquad \textbf{(2)}$$

where α=4/r_a^2 and r_a>0 denotes the neighborhood radius for each cluster center. A data point with many neighboring data points will have a high potential value and the points outside r_a have little in influence on its potential. Note that in Eq. 2, β=4/r_b^2 and r_b>0 presents the radius of the neighborhood for which significant potential revising will occur. After calculating potential for each point, the one with the highest potential value will be selected as the first cluster center. Then the potential of each point is reduced to avoid closely spaced clusters according to Eq. 2. Selecting centers and revising potential is carried out iteratively until a stopping criteria satisfied. If ***max****$_i$*(P_i)≤ε*$P(x_k^*)$, terminate the algorithm, return *k* number of clusters. In this paper, we refer to the subtractive clustering algorithm as ***SC_EstimateK*** in our proposed algorithm for brevity.
To avoid obtaining closely spaced cluster centers, r_b is chosen to be greater than r_a. Typically, r_b =1.5r_a. In step 4, the parameter ε should be selected within (0, 1). If ε is selected to be close to 0, a large number of cluster centers will be generated. On the contrary, a value of ε close to 1 will render fewer cluster centers. In this paper, we utilize subtractive clustering to estimate the approximate number *k* of clusters, which is a reference number of clusters in the next section. For it is approximate estimation, we can set the parameters ε, r_a and r_b to 0.5, 0.5 and 0.75 as default respectively.

3 Cluster Validation Metric and Automatic K-means Clustering algorithm

3.1 Cluster Validation Metric

For k-means clustering algorithm, choosing the number of clusters (*k*) is crucial to the performance of clustering. With variant given the number of clusters, different clustering results may be acquired. In cluster analysis, we find the partitioning that best fits the underlying data through evaluating clustering results. The validity indices are simply and effective methodology of measuring the quality of

clustering results. There are two kinds of validity indices: external indices and internal indices. In order to choose the optimal k value for k-means clustering algorithm, we choose an internal validity index. The principles of some widely-used internal indices for k-estimation and clustering quality evaluation are [15]: Silhouette index [16], Davies-Bouldin index, Calinski-Harabasz index, Dunn index, RMSSTD index. One may choose a validity index to estimate an optimal *k* value, where the optimal clustering solution is found from a series of clustering solutions under different *k* values.
We adapt Silhouette index as cluster validation metric for its simplicity. Silhouette index is a composite index reflecting the compactness and separation of the clusters, and can be applied to different distance metrics. For data instance x_i, its silhouette index $Sil(x_i)$ is definite as:

$$Sil(x_i)=(b(x_i)-a(x_i))/max\{a(x_i),b(x_i)\} \ . \qquad \textbf{(3)}$$

where $a(x_i)$ is the average distance of data instance x_i to other data instances in the same cluster, $b(x_i)$ is the average distance of data instance x_i to instances in its nearest neighbor cluster. The average of $Sil(x_i)$ across all data instances reflects the overall quality of the clustering result. A larger averaged Silhouette index indicates a better overall quality of the clustering result.

3.2 Automatic K-means Clustering Algorithm

K-means clustering [17] is a well-known partitioning method. Objects are classified as belonging to one of *K* groups, *K* computed by the algorithm ***SC_EstimateK*** in section 2. Cluster membership is determined by calculating the centroid for each group (the multidimensional version of the mean) and assigning each object to the group with the closest centroid. This approach minimizes the overall within-cluster dispersion by iterative reallocation of cluster members. We give the following objective function for k-means clustering algorithm:

$$J = \sum_{i=1}^{N}\sum_{k=1}^{K} r_{ik} \|x_i - \mu_k\|^2 \ . \qquad \textbf{(4)}$$

where r_{ik} denotes 1 when x_i belonging to the cluster C_k, 0 when x_i not belonging to the cluster C_k; *N* denotes the number of data instances in data set; *K* denotes *K* clusters in data set; μ_k represents k^{th} cluster centroid, denoted as follows:

$$\mu_k = {}^{1}\!/_{N_k} \sum\nolimits_{x_i \in C_k} x_i \ . \qquad \textbf{(5)}$$

In this paper, we refer to the k-means clustering algorithm as ***K-meansClust*** in our proposed algorithm by brevity. For details of the k-means clustering algorithm, we commend the literature [17].
Firstly, we acquire the proximate number of clusters by the algorithm ***SC_EstimateK*** in section 2. Then we run the algorithm ***K-meansClust*** to find optimal cluster according to the Silhouette index value. The algorithm of finding optimal cluster is as follows:

```
Algorithm: Automatic k-means clustering algorithm (AK-meansClust)
Input: A data set $D=\{x_1,x_2,\ldots,x_N\}$;
Output: Optimal clusters $Clust_{opt}=\{C_1,C_2,\ldots,C_{opt}\}$;
Step 1. $K$=SC_EstimateK ($D$)
Step 2. for i=$K$-λ:$K$+λ do loop
Step 3.  $Clust[i]$=K-meansClust($D,i$)
Step 4.  $S[i]=Sil(Clust[i])$
Step 5.  $Array$[i]=$\{S[i], Clust[i]\}$
Step 6. end_loop
   Step 7. Find maximal Silhouette index value and corresponding cluster in
           $Array$, and the found index is denoted as $maxindx$.
Step 8. Return $Clust_{opt}=Clust[maxindx]$;
```

In step 2, ***AK-meansClust*** algorithm run 2*λ times loop and produce the clusters of data set and corresponding Silhouette index value for every time, where λ is a desired value and can be a constant or adjusts to K value. In this paper, for the sake of simplicity, λ is set to $K/2$.

4 Improved CBLOF and Outlier Detection Algorithm

In this section, we take use of the local outlier factor to identify the top n outliers in data set by analyzing the clustering structure obtained in section 3. It is reasonable to define the outliers based on the structure of clusters and identify those objects that do not lie in any large clusters as outliers. As the large clusters are often dominant in the data set, the outliers are less probably included in them. Before presenting our outlier detection algorithm, we describe the following definition about the local outlier factor [12].

Definition 1. (large and small cluster) Suppose that $C=\{C_1, \ldots, C_k\}$ is the set of clusters in the sequence that $|C_1| \geq \ldots \geq |C_k|$, where $|C_i|$ denotes the number of objects in C_i (i=1,…,k) and k is the number of clusters. Given two numeric parameters α and β, we define b as the boundary of large and small cluster if one of the following formulas holds.

$$|C_1|+\ldots+|C_b| \geq |D|*\alpha. \quad (6)$$

$$|C_b|/|C_{b+1}| \geq \beta. \quad (7)$$

Then, the set of large cluster is defined as $LC=\{C_i|i \leq b\}$ and the set of small cluster is defined as $SC=\{C_j|j>b\}$.

Definition 1 provides quantitative measure to distinguish large and small clusters. Formula 6 considers the fact that most data points in the data set are not outliers. Therefore, clusters that hold a large portion of data points should be treated as large clusters. Formula 7 considers the fact that large and small clusters should have significant differences in size.

To study clustering algorithm across different data set with outliers, we find that fact the point outliers will be classified as the nearest large clusters, since point outlier can not be clustered as one cluster. Therefore, the design of a new local outlier factor is desired in this situation. We designed an improved cluster-based local outlier factor based on the definition in the previous studies [12], [18].

Definition 2. (Cluster-Based Local Outlier Factor) Suppose $C=\{C_1, \ldots, C_k\}$ is the set of clusters in the sequence that $|C_1| \geq \ldots \geq |C_k|$ and the meanings of α, β, b, LC and SC are the same as they are formalized in Definition 1. For any record t, the cluster-based local outlier factor of t is defined as:

$$CBLOF(t)=\begin{cases} dist(t,C_j)/|C_i|, \text{where } t \in C_i, C_i \in SC \text{ and } C_j \in LC \\ dist(t,C_j)\Big/\left(\sum_{C_j \in LC} |C_j| \Big/ |LC|\right), \text{where } t \in C_j \text{ and } C_j \in LC \end{cases} \quad \mathbf{(8)}$$

where $dist(t,C_j)$ is the Euclidean distance between t and the center of C_j, $|C_j|$ is the number of objects in C_j, $|LC|$ is the number of clusters in the large clusters set. ***CBLOF***(t) is simply to be calculated because we just need know the number of objects in certain large clusters as well as how far it close to t.

If $t \in SC$, C_j denotes the nearest large cluster which is neighboring to t. In this case, $\forall t \in SC$, C_j is almost identical. Thus, the object t which is more far away from the center of C_j will get a larger ***CBLOF***. Otherwise, if $t \in LC$, C_j denotes the large cluster which contains t. Assume that two objects are respectively within a large cluster and a small cluster, they have equal distance from their corresponding center of clusters. In this situation, the object t belong to the small cluster will get a larger ***CBLOF*** because we consider that data points in small clusters are outliers and provide more meaningful outlying information than point outliers in large clusters. In addition, in a sense, we can regard point outliers as noises and they provide less underlying outlying information about the data set.

Based on the clusters obtained in section 3, we can give an outlier detection algorithm.

```
Algorithm: Automatic K-means clustering algorithm for outlier detection
(AKCOD)
Input: A data set $D=\{x_1,x_2,\ldots,x_N\}$; parameters $n$, $\alpha$ and $\beta$
Output: The top $n$ outliers with largest values of $CBLOF$ in data set
   Step 1. Utilize AK-meansClust algorithm to produce optimal cluster $Clust_{opt}$=
           $\{C_1,C_2,\ldots,C_{opt}\}$;
Step 2. Get $LC$ and $SC$ according to Definition 2 based on parameters $\alpha$ and $\beta$;
   Step 3. For each object $t$ in the data set, calculate CBLOF($t$) according to Eq.
           8;
   Step 4. Return the top $n$ outliers with largest CBLOF
```

5 Experiments

In this section, we conduct extensive experiments to evaluate the performance of our proposed algorithms. All algorithms are implemented through MATLAB on Pentium(R) Dual-Core CPU 2.0G PC with 2.0G main memory. We utilize the synthetic data set with outliers and real life data sets obtained from the UCI machine learning repository [21] to compare our algorithm against the original CBLOF algorithm (ORCBLOF) [18], KNN [10] and LOF [11] for outlier detection. For the results of KNN and LOF algorithm, we only present the best overall performance with the optimal number of neighbors. For ORCBLOF algorithm, we choose the same parameters with our proposed algorithm.

We evaluate outlier detection techniques using two categorical different metrics. The first metric consists of detection rate (denoted as DR) and false alarm rate (denoted as FR) [19], which is the most commonly used in detection systems, defined as follows:

$$DR=|AO|/|CO| \ . \qquad \textbf{(9)}$$

$$FR= (|BO|-|AO|)/(|DN|-|CO|) \ . \qquad \textbf{(10)}$$

where $|AO|$ is the number of true outliers in the detected top n outliers, $|BO|$ is the number of the detected top n outliers and $|CO|$ is the number of true outliers in the entire data set; while $|DN|$ is the number of all objects in the data set.

One drawback of the above type metric is that it is highly dependent on the choice of n, which is used for top n outliers. An outlier detection technique might show 100% DR for a particular value of n but show 50% accuracy for $2n$. To overcome this drawback we also use the following evaluation metric. The second categorical metric used to evaluate the outlier detection techniques is to obtain the area under the ROC curve (AUC) [20] obtained by varying n from 1 to $|DN|$. The advantage of AUC is that it is not dependent on the choice of n.

5.1 Effectiveness of Detecting Outliers on Synthetic Data Set

Figure 47.1 shows a synthetic 2-dimentional data set with three large clusters, two small clusters and four point outliers. Intuitively, all of 22 objects in the small clusters and 4 point outliers denoted with "*" (O_1 to O_6) can be regards as outliers. We apply ***AKCOD***, ORCBLOF, KNN and LOF to find the top 26 outliers in the dataset. ***AKCOD*** can successfully find the desirable outliers because it utilizes ***AK-meansClust*** algorithm to obtain stable clusters and effectively defines the local outlier factor in both large and small clusters for every object. ORCBLOF is not able to find outliers which are near to enormous clusters. According to the definition in [18], the skewed enormous clusters would decrease CBLOF value of outliers near to enormous clusters. For example, O_1 and O_2 will be assigned smaller CBLOF values relative to normal objects in C_3 and be treated as normal objects. However, LOF has some difficulty to distinguish outliers since the density of C_3 and O_6 are similar. So it identifies some normal objects in C_3 as outliers rather than detects the outlier cluster O_6. On the other hand, KNN fails to identify outlier cluster O_5 because it is much closed to C_3. In table 47.1, we present the performance of outlier detection algorithms adapted in this paper.

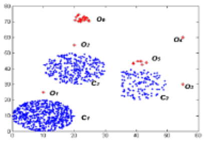


Fig. 47.1. 2-dimensional data set with outliers

Table 47.1. Results of outlier detection algorithm on synthetic data set.

	KNN	LOF	ORCBLOF	***AKCOD***
DR	0.75	0.85	0.91	1
FR	0.04	0.002	0.001	0
AUC	0.96	0.98	0.99	1

5.2 Effectiveness of Detecting Outliers on Real Life Data Set

To verify the performance of our proposed algorithm on practical application domain, we apply all algorithms on several different real life data sets available at the UCI Machine Learning Repository. The details about the selected data sets are summarized in Table 47.2. For the sake of simplicity, we choose all data sets with purely continuous attributes. For the mixture of continuous and categorical attributes, a possible way is to compute the similarity for continuous and categorical attributes separately, and then do a weighted aggregation.

The "No. attributes" in table 47.2 denotes all attributes except the class attribute. Note that the "No. outliers" in the table is the total of objects in the small clusters. Each data set contains labeled instances belonging to multiple classes. We identify the last class or last two classes as the outlier class, and rest of the classes were grouped together and called normal. In order to produce the small clusters, we remove most objects within the last class and set the proportion of outliers to be not greater than 10% of total instances in data set. For Lymph data set, we choose two class containing least instances as outlier classes. In the experiments, the parameters α and β set to 0.75 and 5 respectively.

Tables 47.3 summarize the DR, FR and AUC results on the real life data sets. ***AKCOD*** algorithm performs munch better than ORCBLOF, LOF and KNN with a lower false alarm rate and higher AUC. While running on Optical data set, KNN performs extremely poorly because the data set is high dimensional and very spare and in turn identify the outliers simply based on Euclidean distance is not feasible. Though ***AKCOD*** performs moderately on Optical data set, it still outperforms both LOF and KNN.

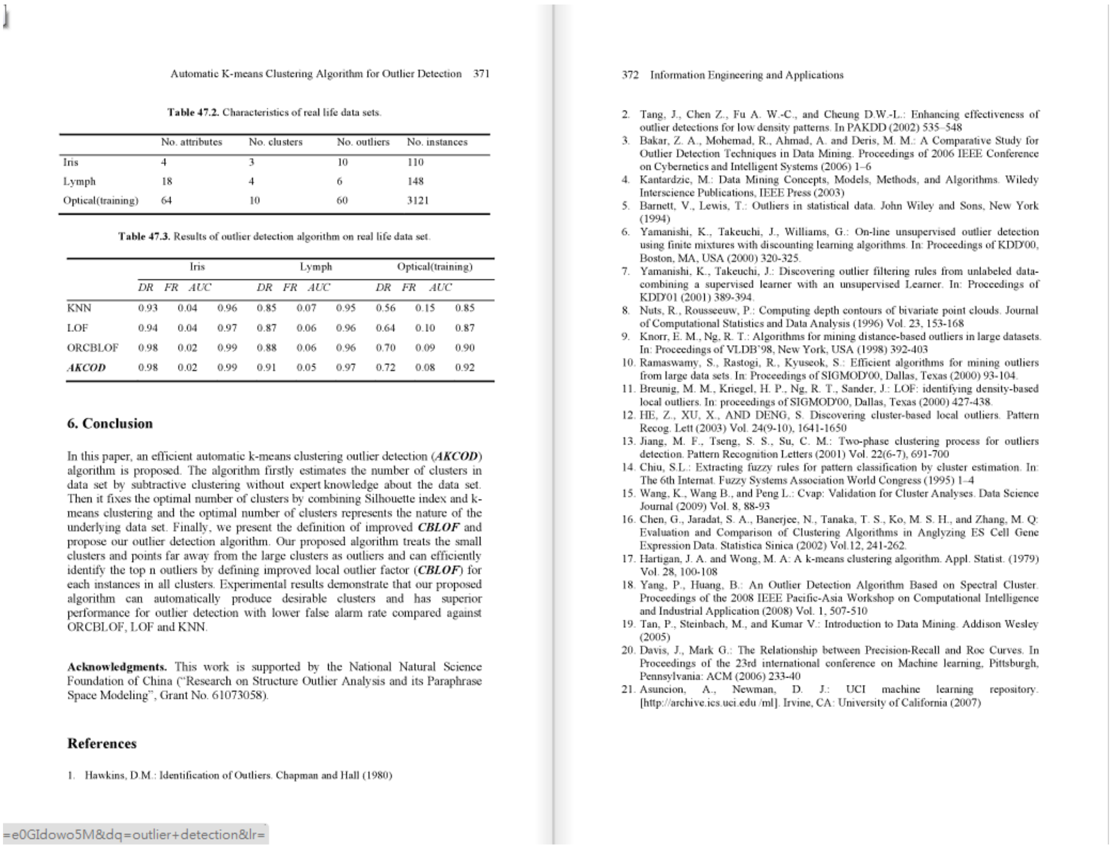


Table 47.2. Characteristics of real life data sets.

	No. attributes	No. clusters	No. outliers	No. instances
Iris	4	3	10	110
Lymph	18	4	6	148
Optical(training)	64	10	60	3121

Table 47.3. Results of outlier detection algorithm on real life data set.

	Iris			Lymph			Optical(training)		
	DR	*FR*	*AUC*	*DR*	*FR*	*AUC*	*DR*	*FR*	*AUC*
KNN	0.93	0.04	0.96	0.85	0.07	0.95	0.56	0.15	0.85
LOF	0.94	0.04	0.97	0.87	0.06	0.96	0.64	0.10	0.87
ORCBLOF	0.98	0.02	0.99	0.88	0.06	0.96	0.70	0.09	0.90
AKCOD	0.98	0.02	0.99	0.91	0.05	0.97	0.72	0.08	0.92

6. Conclusion

In this paper, an efficient automatic k-means clustering outlier detection (***AKCOD***) algorithm is proposed. The algorithm firstly estimates the number of clusters in data set by subtractive clustering without expert knowledge about the data set. Then it fixes the optimal number of clusters by combining Silhouette index and k-means clustering and the optimal number of clusters represents the nature of the underlying data set. Finally, we present the definition of improved ***CBLOF*** and propose our outlier detection algorithm. Our proposed algorithm treats the small clusters and points far away from the large clusters as outliers and can efficiently identify the top n outliers by defining improved local outlier factor (***CBLOF***) for each instances in all clusters. Experimental results demonstrate that our proposed algorithm can automatically produce desirable clusters and has superior performance for outlier detection with lower false alarm rate compared against ORCBLOF, LOF and KNN.

Acknowledgments. This work is supported by the National Natural Science Foundation of China ("Research on Structure Outlier Analysis and its Paraphrase Space Modeling", Grant No. 61073058).

References

1. Hawkins, D.M.: Identification of Outliers. Chapman and Hall (1980)

2. Tang, J., Chen Z., Fu A. W.-C., and Cheung D.W.-L.: Enhancing effectiveness of outlier detections for low density patterns. In PAKDD (2002) 535–548
3. Bakar, Z. A., Mohemad, R., Ahmad, A. and Deris, M. M.: A Comparative Study for Outlier Detection Techniques in Data Mining. Proceedings of 2006 IEEE Conference on Cybernetics and Intelligent Systems (2006) 1–6
4. Kantardzic, M.: Data Mining Concepts, Models, Methods, and Algorithms. Wiledy Interscience Publications, IEEE Press (2003)
5. Barnett, V., Lewis, T.: Outliers in statistical data. John Wiley and Sons, New York (1994)
6. Yamanishi, K., Takeuchi, J., Williams, G.: On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. In: Proceedings of KDD'00, Boston, MA, USA (2000) 320-325.
7. Yamanishi, K., Takeuchi, J.: Discovering outlier filtering rules from unlabeled data-combining a supervised learner with an unsupervised Learner. In: Proceedings of KDD'01 (2001) 389-394.
8. Nuts, R., Rousseeuw, P.: Computing depth contours of bivariate point clouds. Journal of Computational Statistics and Data Analysis (1996) Vol. 23, 153-168
9. Knorr, E. M., Ng, R. T.: Algorithms for mining distance-based outliers in large datasets. In: Proceedings of VLDB'98, New York, USA (1998) 392-403
10. Ramaswamy, S., Rastogi, R., Kyuseok, S.: Efficient algorithms for mining outliers from large data sets. In: Proceedings of SIGMOD'00, Dallas, Texas (2000) 93-104.
11. Breunig, M. M., Kriegel, H. P., Ng, R. T., Sander, J.: LOF: identifying density-based local outliers. In: proceedings of SIGMOD'00, Dallas, Texas (2000) 427-438.
12. HE, Z., XU, X., AND DENG, S. Discovering cluster-based local outliers. Pattern Recog. Lett (2003) Vol. 24(9-10), 1641-1650
13. Jiang, M. F., Tseng, S. S., Su, C. M.: Two-phase clustering process for outliers detection. Pattern Recognition Letters (2001) Vol. 22(6-7), 691-700
14. Chiu, S.L.: Extracting fuzzy rules for pattern classification by cluster estimation. In: The 6th Internat. Fuzzy Systems Association World Congress (1995) 1–4
15. Wang, K., Wang B., and Peng L.: Cvap: Validation for Cluster Analyses. Data Science Journal (2009) Vol. 8, 88-93
16. Chen, G., Jaradat, S. A., Banerjee, N., Tanaka, T. S., Ko, M. S. H., and Zhang, M. Q: Evaluation and Comparison of Clustering Algorithms in Anglyzing ES Cell Gene Expression Data. Statistica Sinica (2002) Vol.12, 241-262.
17. Hartigan, J. A. and Wong, M. A: A k-means clustering algorithm. Appl. Statist. (1979) Vol. 28, 100-108
18. Yang, P., Huang, B.: An Outlier Detection Algorithm Based on Spectral Cluster. Proceedings of the 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application (2008) Vol. 1, 507-510
19. Tan, P., Steinbach, M., and Kumar V.: Introduction to Data Mining. Addison Wesley (2005)
20. Davis, J., Mark G.: The Relationship between Precision-Recall and Roc Curves. In Proceedings of the 23rd international conference on Machine learning, Pittsburgh, Pennsylvania: ACM (2006) 233-40
21. Asuncion, A., Newman, D. J.: UCI machine learning repository. [http://archive.ics.uci.edu /ml]. Irvine, CA: University of California (2007)