ELSEVIER

Pattern Recognition Letters 22 (2001) 691–700

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Two-phase clustering process for outliers detection

M.F. Jiang, S.S. Tseng *, C.M. Su

Department of Computer and Information Science, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu 30050, Taiwan, ROC

Received 17 November 1999; received in revised form 25 April 2000

Abstract

In this paper, a two-phase clustering algorithm for outliers detection is proposed. We first modify the traditional *k*-means algorithm in Phase 1 by using a heuristic "if one new input pattern is far enough away from all clusters' centers, then assign it as a new cluster center". It results that the data points in the same cluster may be most likely all outliers or all non-outliers. And then we construct a minimum spanning tree (MST) in Phase 2 and remove the longest edge. The small clusters, the tree with less number of nodes, are selected and regarded as outlier. The experimental results show that our process works well. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Outliers; *k*-means clustering; Two-phase clustering; MST

1. Introduction

Cluster analysis could be defined as the process of separating a set of patterns (or objects) into clusters (or groups) such that members of one cluster are similar. The goal of such partitioning, or clustering, may be to gain an insight into some structure inherence in the population (such as subspecies grouping of plants or animals) or to develop a business strategy that is customized to each cluster customer for higher business efficiency.

The definition of the term "remainder cluster" (Cedno and Suer, 1997) is used to designate few patterns that are much different from other clusters. They are often to be seemed as noises like outliers. Among all data, remainder cluster often has small proportion and each parameter value differs greatly from other normal data. Therefore, in most traditional clustering algorithms, the patterns are either neglected or given a lower weight to avoid other data being clustered.

However, in some specific applications we have to find out the anomaly patterns from a large amount of data (Su et al., 1999). For examples, in the medicine domain, we may want to find extraordinary cases of patient data; in network problem domain, we may want to find the anomaly behaviors from log data to prevent some anomaly attacks. Identifying outlying observations is also an important aspect of the regression model-building process. Outlying observations should be identified because of their potential effect on the fitted model. That is, occasionally certain observations will have a disproportionate effect on the parameter estimates, precision of estimated parameters, and the overall predictive ability of the model.

As we know, if we partition 50 data points into five clusters it will have 7.4×10^{32} possibilities

* Corresponding author. Tel.: +886-3-5731-966, fax: +886-3-5721-490.

E-mail address: sstseng@cis.nctu.edu.tw (S.S. Tseng).

PII: S0167-8655(00)00131-8

(Kaufman and Rousseeuw, 1990). The remainder clusters are often too small to be classified. So using traditional clustering algorithm to cluster the minor and anomaly patterns will either cost much time or not work well. In the real case, the data set often have more than ten thousand records, the problem mentioned in the previous paragraph may not be easily solved by traditional clustering algorithms.

In this work, the outlier is defined as the smaller clusters which are far from the most of points. Our idea is first to partition the data points into several clusters each of which may be all outliers or all non-outliers. That is, when the points contained in the same cluster are not closer enough, the cluster may be split into two smaller clusters. After partitioning the data points, it can be easily seen that the time complexity for finding the outliers clusters may be reduced. This is because the similar points are merged into the same cluster, and the data to be processed are clusters instead of points. In this paper, a two-phase clustering algorithm for outliers detection is proposed. In the first phase, we propose a modified *k*-means algorithm by using a heuristic "if one new input pattern is far enough away from all clusters' centers, then assign it as a new cluster center". It results that the diameter of each cluster may be reduced and the data points in the same cluster may be most likely all outliers or all non-outliers. But the side effect, the number of clusters obtained in this phase more than that obtained in original *k*-means algorithm, may probably be caused. And then we propose an outlier-finding process (OFP) in the second phase to find the outliers from the resulting clusters obtained in Phase 1. By the process, a minimum spanning tree (MST) (Zahn, 1971) for these clusters is first constructed and treated as a member of a forest. Then remove the longest edge of a tree from the forest and replace the original tree with two newly generated subtrees. The small clusters, the tree with less number of nodes, are selected and regarded as outlier. Furthermore, we may repeatedly remove the longest edge from the forest until the number of trees is sufficient enough.

The rest of this paper is organized as follows. Section 2 presents the related works of clustering and outlier detection. Section 3 gives a detailed description of our algorithm. In Section 4, we present some experimental results on different data to show the performance of our algorithm. Concluding remarks are given in Section 5.

2. Related works

Clustering techniques have received attention in many areas, such as engineering, medicine, biology and computer science. The purpose of clustering is to group together data points that are close to one another (Tou and Gonzalez, 1974). Kaufman and Rousseeuw (1990) referred to clustering as the "art of finding groups in data". There exists no best clustering procedure because the success of clustering technique is highly dependent on the type of data and the particular purpose of the investigation. Although recent research has attempted to make cluster analysis more statistically oriented, it is still largely regarded as an exploratory tool.

Observations that do not follow the same model as the rest of the data are typically called outliers. In many fields like machine learning, image recognition, clustering, etc., the experimental data including outliers or noises may cause bad effects of results. In this sense, appropriate model selection is synonymous with absence of outliers in the data. In many approaches for the selection of variables and identification of outliers, several robust algorithms have been proposed for detecting multiple outliers (Hadi and Simonoff, 1993; Kianifard and Swallow, 1990) which can produce estimates by giving up the outliers for the purpose of inference.

There are various clustering methods making their classification a difficult task. Among clustering methods, three famous algorithms, *k*-means algorithm, hierarchical clustering method, and graph clustering theory, which are related to our new method, are briefly described as follows.

2.1. k-means algorithm

The process of *k*-means clustering (Forgy, 1965; McQueen, 1967) searches for a nearly optimal

partition with a fixed number of clusters. First an initial partition with the chosen number of clusters is built (it can be done in many ways). Then, keeping the same number of clusters, the partition is improved iteratively. In the rest of this paper, assume there are m patterns to be clustered. Each pattern is handled sequentially and reassigned to the cluster such that the partitioning criteria is most improved by the reassignment. Usually the procedure ends when no improving reassignment is obtained, but stronger end tests can also be used.

Let the set of patterns M be $\{x_1, x_2, \ldots, x_m\}$ for $x_i \in R^n$, where x_i is the ith pattern. Let the number of target clusters be k. The iterative clustering algorithm (Forgy, 1965) is as below:

Step 1. Select an initial partition with k clusters.

Step 2. Generate a new partition by assigning each pattern to its closest cluster's center.

Step 3. Compute new cluster center as the centroid of the clusters.

Step 4. Repeat Steps 2 and 3 until an optimum value of the criteria function is found.

Step 5. Adjust the number of clusters by merging and splitting existing clusters or by removing outlier clusters.

A k-means method as an assignment of all patterns to the closest cluster's center is defined in Step 2, and Step 3 is the process of updating the partitions. Many methods determining initial partition are suggested by Hyvarinen (1963), Forgy (1965), McQueen (1967), Ball and Hall (1964), etc. Besides, there are two different ways of updating the partitions in k-means method. In McQueen's k-means method, the center of the gaining cluster is recomputed after each new assignment. Forgy's method re-computes clusters' centers after all patterns have been examined. Our clustering method is to modify the McQueen's method by adjusting the centroids after allocating, splitting, or merging operations. For the inappropriate initial partition, some modified algorithms, e.g., ISODATA (Ball and Hall, 1964), can create new clusters or merge existing clusters if certain conditions are met. A cluster is split if it has too many patterns, and two clusters are merged if their cluster centers are sufficiently close.

Johnson and Wichern (1982) noted that the Euclidean distance is the most widely accepted and commonly used measure of similarity when trying to find groups among multivariate patterns. Euclidean distances, based on Pythagorean's theorem, are the square root of sum-of-squares of differences between patterns. Consider the patterns $x_1 = (x_{11}, x_{12}, \ldots, x_{1p})$ and $x_2 = (x_{21}, x_{22}, \ldots, x_{2p})$, where x_{ij} represents the value for the ith pattern on the jth variable. The Euclidean distance, d, between x_1 and x_2 is

$$d(x_1, x_2) = \sqrt{(x_{11} - x_{12})^2 + (x_{12} - x_{22})^2 + \cdots + (x_{1p} - x_{2p})^2}.$$

The Euclidean distance is popular because of its intuitive appeal as a similarity measure. That is, similar observations should be separated by a relatively small distance, while dissimilar observations should be separated by a relatively large distance.

2.2. Hierarchical clustering method

The hierarchical algorithms form an initial partition of m clusters (each pattern is seen as a cluster) and then proceed to reduce the number of clusters one at a time until all m observations are in a single cluster (Dubes and Jain, 1987). The difference between various clustering techniques is the rule for merging clusters. Single linkage merges clusters based on the distance (''similarity measure'') between the two closest observations in each cluster. Because of this, single linkage is commonly referred to as the ''nearest neighbor'' algorithm. For more details on the single linkage clustering algorithm and other clustering algorithms, refer to Johnson and Wichern (1982).

The results of the single linkage clustering algorithm can be seen on a diagram, or what is commonly referred to as a ''cluster tree''. Specifically, the cluster tree must be partitioned or ''cut'' at a certain height. The number of groups depends upon where the tree is cut. The ''number of groups'' problem is a practical issue that any user of clustering procedures must deal with.

Hierarchical clustering model, which is often using the single linkage measurement to cluster the points, provides a way to determine the number of groups. In hierarchical clustering model, single

linkage merges clusters based on the distance between the two closest observations in each cluster, and the results of the clustering can be seen as "cluster tree". The clusters which are added later can be seen as outliers or remainder clusters. However, it is not suitable for large amount of data, since the computational cost of single linkage using modern methods is $O(n^2)$ for CPU time and $O(n)$ for memory space, or vice versa, for n patterns to be clustered (Dubes and Jain, 1987).

2.3. Graph clustering theory

A graph is a mathematical structure that has a multitude of applications in cluster analysis. A graph can be defined as a set of vertices and edges. The vertices $V = \{v_i\}$ which usually represent the objects being clustered. A set of edges $E = \{e_j\}$ records the interactions between pairs of vertices. In the present work we consider the particular case when the graph is a tree $T = (V, E)$. Clustering problems on trees arise in a variety of applications. Trees are used to represent hierarchical databases, and many communication or distribution networks display a tree-like structure.

In the case of a tree, the feasible partitions can be characterized as follows. If one arbitrarily "cuts" (i.e. deletes) $p-1$ edges from a tree, p disjoint subtrees $T_1, \ldots, T_p$ are obtained. If C_k is the vertex-set of $T_k (k = 1, \ldots, p)$, then $\pi = \{C_1, \ldots, C_p\}$ is a feasible partition of V into p clusters. As a matter of fact, there are exactly $p-1$ outer edges with respect to π by cutting them to obtain π.

Clustering problems on trees can be solved in polynomial time by dynamic programming for virtually all the objective functions used in practice. Clustering problems on general graphs tend to be NP-complete; it turns out that the case of trees is borderline and the computational complexity is influenced by the objective function.

3. Two-phase clustering process

In data-mining applications where data tend to be very large, e.g., 10 million records or more, an important factor is the computing resources required (CPU time and memory space). Resource consumption varies considerably between the different methods, and some methods become impractical for all but small data sets, while the k-means algorithms generally have a time and space complexity of order n, where n is the number of records in the data set. In this section, a two-phase clustering algorithm for outliers is proposed. In first phase, k-means algorithm is modified to cluster a large quantity in a coarse manner allowing the amount of clusters more than the original k where k is the requested number of clusters including outliers set by user. In the modified k-means algorithm, we use the heuristic "if one new input pattern is far enough away from all clusters' centers, then assign it as a new cluster center". It results that the diameter of each cluster may be reduced and the data points in the same cluster are closer to one another. By this method, the data points in the same cluster may be all outliers or all non-outliers, but the side effect is that the number of clusters found in this phase is more than that found in original k-means algorithm. In order to find the outliers, in the second phase, an MST is built according to the distance of edges where each node of the tree represents the center of each cluster obtained in Phase 1. Then the longest edge is removed and the original tree is replaced by two newly generated subtree. The small clusters, the trees with less number of nodes, are selected and regarded as outlier. Furthermore, we may repeatedly remove the longest edges until the number of clusters is sufficient enough and the left clusters are not much balance partitioned as traditional k-means algorithm does. The algorithm of our clustering method is divided into two phases, modified k-means process (MKP phase) and OFP phase, are described in the following sections.

3.1. Modified k-means process (MKP)

Since different clustering results may be generated for different initial data set, the desired optimal result cannot be easily obtained by using k-means algorithm. Especially in a large number of patterns, it is impracticable to find the global optimal case by trying all different initial sets. Our modified k-means algorithm uses a heuristic "if

one new input pattern is far enough away from all clusters' centers, then assign it as a new cluster center'', which is splitting the outlier as another cluster center in cluster iteration process. This heuristic is permitted to adjust the number of clusters. Similar to ISODATA algorithm, adding more clusters will help us to find out potential outliers or remainder cluster which is not conspicuous.

At the beginning our clustering method is like traditional k-means algorithm. Let k' be the number of adjustable clusters. Initially, set $k' = k$, and randomly choose k' patterns as cluster centers, $C = \{z_1, z_2, \ldots, z_{k'}\}$ in R^n. In the period of iteration, when adding one pattern to its nearest cluster, we compute not only the new cluster's center but also the minimum distance $\min(C)$ between any two clusters' centers.

$$\min(C) = \min ||z_j - z_h||^2 \quad \text{for } j, h = 1, \ldots, k',\ j \neq h.$$

For any pattern x_i the minimum distance $\min(x_i, C)$ of its nearest cluster's center is computed by

$$\min(x_i, C) = \min ||x_i - z_j||^2 \quad \text{for } j = 1, \ldots, k'.$$

According to the above heuristic, more clusters can be found after splitting the data. In some extreme case, each pattern is put into its own cluster, i.e., $k' = m$, where m is the number of all points, and this will result in too many clusters. So the number $k_{\max}$ is defined as the maximum number of cluster we may afford. In the modified k-means algorithm, when the patterns are split into $k_{\max} + 1$ clusters, two closest clusters will be merged.

The modified k-means algorithm is as follows:

Algorithm 3.1 (*Modified k-means process*, *MKP*).

Step 1. Randomly choose k' initial seeds as cluster centers.

Step 2. For $i \leftarrow 1$ to m do compute $\min(C)$ and $\min(x_i, C)$.

Step 3. If $\min(x_i, C) \leqslant \min(C)$ then go to Step 6.

Step 4. SPLITING PROCESS: Assign x_i to be the center of a new cluster $C_{k'}$ and set $k' \leftarrow k' + 1$.

Step 5. MERGING PROCESS: If $k' > k_{\max}$, merge the two closest clusters into one cluster and set $k' = k_{\max}$.

Step 6. ALLOCATING PROCESS: Assign x_i to its nearest cluster.

Step 7. Go to Step 2 until the cluster membership stabilized.

In Step 2, the distance between ''the distance between the pattern and the nearest cluster'' and ''the minimum distance among existing clusters'' is computed, and in Step 3, the heuristic ''if one pattern is far enough, then assign it as a new cluster'' is used to determine which one should be split or merged. It is important to note that the splitting and merging operation employed here are quite similar to the splitting and merging operations proposed in ISODATA algorithm. However, unlike ISODATA algorithm, it sets parameters by user to determine splitting when a cluster has too many patterns or merging if two clusters' centers are sufficiently close. It is not suitable when user knows neither the property nor the scatter of the patterns. In our scheme splitting performs due to the condition at that time, and merging performs when too many clusters occur. It is not necessary to thoroughly know the distribution of the data before clustering. The value of $k_{\max}$ is ranged from k to m and determined by the domain expert. If the data points are random, we suggest that the value of $k_{\max}$ is set near m.

(a) MERGING PROCESS

The process of merging is to lump two existing clusters. Set z_{k1} and z_{k2} to be the centers of the two closest clusters for merging. Thereafter, these two clusters are merged by following operation:

$$z_k^* = \frac{1}{N_{k1} + N_{k2}} [N_{k1} z_{k1} + N_{k2} z_{k2}],$$

where z_k^* is the center of the new cluster, and N_{k1}, N_{k2} are the populations of the two clusters. Clearly, z_{k1} and z_{k2} are replaced by z_k^*, and the number of clusters k' is reduced by one.

(b) ALLOCATING PROCESS

For each clustering iteration, we allocate all patterns to the closest cluster, such that the sum of squared Euclidean distances between each pattern and the center of the corresponding cluster are minimized. The total criteria can be described as to minimize: $j(w, z) = \sum_{i=1}^{m} \sum_{j=1}^{k'} w_{ij} \|x_i - z_j\|^2$ subject to $\sum_{j=1}^{k} w_{ij} = 1, i = 1, 2, \ldots, m$, where $w_{ij} = 1$, if

pattern i is allocated to cluster j for all $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, k$ or $w_{ij} = 0$, otherwise.

Example 1. For the patterns {1,2, 3, 4, 6, 10 }, the results by our modified k-means algorithm are showed in Table 1.

Since the relative performances of k-means and the outlier detection algorithm may well vary with a range of k, a criteria function is defined as: $(1/ARAC)$, where ARAC (average relation among clusters) is a relation criteria of one cluster with other clusters which is similar to Davies-Bouldin index (Davies and Bouldin, 1979). When the ARAC become smaller, the criteria function will get positive grow. By comparing the criteria values of different cluster number, we can find out the optimum number of clusters.

3.2. Outlier-finding process (OFP)

Although most cluster techniques use density as criteria to determine the result of clustering is good or not, it seems unnecessary in our method to split a large cluster into several partitions for reducing the density. Similar to the criteria of hierarchical techniques, the criteria of our method to find outlier data is determined with distances. Therefore, in Phase 2, k' clusters' centers obtained in Phase 1 will be regarded as k' nodes which will be used to construct an MST based upon the distance between every two nodes. The reason why we construct the MST instead of hierarchical one is that the time complexity of constructing MST is $O(n^2)$ but hierarchical one is $O(n^3)$ (Jain and Dubes, 1988).

In Phase 2, we propose an OFP to find the outliers from the resulting clusters obtained in Phase 1. In Step 1, we first construct an MST for these clusters and set it as a member of a forest F. Then in Step 2, we remove the longest edge of a tree from the forest and replace the tree with two newly obtained subtrees. The small clusters, the tree with less number of nodes, are selected and regarded as outlier. Furthermore, we may repeatedly remove the longest edge from the forest until the number of trees in F equals to k.

Let the input of the process be centroids set $C = \{z_1, z_2, \ldots, z_{k'}\}$ in R^n, and assume an empty tree set F is initialized. The OFP is as follows:

Algorithm 3.2 (*Outlier-finding process*, *OFP*).

Step 1. Construct an MST by the centroids of set C and add it to F.

Step 2. Remove the longest edge from the tree in F and replace the tree with two newly obtained subtrees.

Step 3. Regard the clusters in the small subtree as outliers.

4. Experiments

In this section, three different experimental data are tested to compare our new method with traditional clustering algorithms. In the first experiment, we use 150 two-dimensional data from Iris flower data. In the second experiment, the four-dimensional sugar-cane breeding data set is tested. Finally, we apply our process in network behavior detection. All the experimental results show that our outlier-detecting process generally works better than traditional k-means algorithms.

Table 1
The results by our modified k-means algorithm

	Cluster 1	Cluster 2	Cluster 3	Minimum distance
Initial set	{}:1	{}:2		1
Allocate {1}	{1}:1	{}:2		1
Allocate {2}	{1}:1	{2}:2		1
Allocate {3}	{1}:1	{2,3}:2.5		1.5
Allocate {4}	{1}:1	{2,3,4}:3		2
Allocate {6}	{1}:1	{2,3,4}:3	{6}:6	2 (split)
Allocate {10}	{1,2,3,4}:2.5	{6}:6	{10}:10	3.5 (merge)

4.1. Iris data of two attributes

The first part of this experiment involves data supplied by well-known Iris flower data (Duda and Hart, 1973). There are 3 classes of Iris in Iris Plants Database: Iris-setosa, Iris-versicolor, and Iris-viginica. It is known that Iris data have some noises which can be overcome by previous methods. In this experiment, we choose 1st and 2nd attributes as two input parameters. Fig. 1(a) and (b) show the results of k-means algorithm with $k=3$ and $k=4$, respectively; Fig. 1(a′) and (b′) show the result of our outliers clustering process with $k=3$ and $k=4$, respectively.

In the second part of the experiment, we choose 2nd and 3rd attributes of Iris as input parameters. Similarly, Fig. 2(c) and (d) are obtained by k-means algorithm with $k=3$ and $k=4$, respectively, and Fig. 2(c′) and (d′) are obtained by our outliers clustering with $k=3$ and $k=4$, respectively. According to the result, it can be easily seen that k-means algorithm cluster patterns in a balance way while our cluster process would not partition a large cluster into several subclusters. On the contrary, our process would cluster the few patterns on the edge of distribution which match our goal of clustering outliers.

4.2. Sugar-cane breeding data

To understand the abilities of the outliers detection algorithm, the sugar-cane breeding database (Jiang et al., 1996) established by Taiwan Sugar Research Institute (TSRI) in 1990 is tested. Each data entry includes the name, 23 kinds of characteristics of sugar-cane such as stalk diameter, stalk length, etc. In the experiment, 171 entries and four features, stalk diameter (DIA), stalk length (LGH), enhancing sugar (ESG), and flower percentage (FPC) are used. After clustering, as shown in Tables 2 and 3, 3 clusters are generated and two outliers are detected with $k=5$.

4.3. E-mail log data

Spam usually means that a great quantity of mails are sent in a short time to the mail server, which will increase the work load of the mail server and crash down the whole system. Thus we develop an on-line monitoring of anomaly E-mail

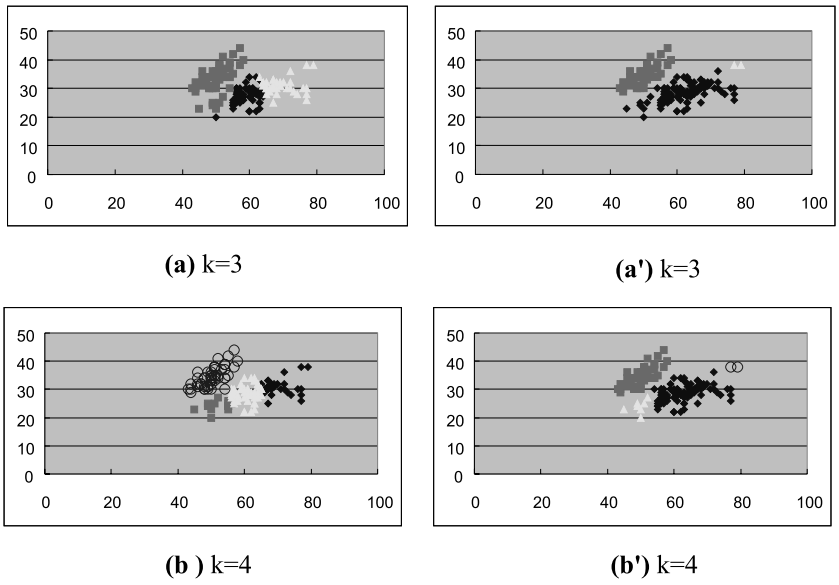

Fig. 1. The results of first experiment. (a, b) are obtained by k-means algorithm while (a', b') are obtained by our outliers clustering process.

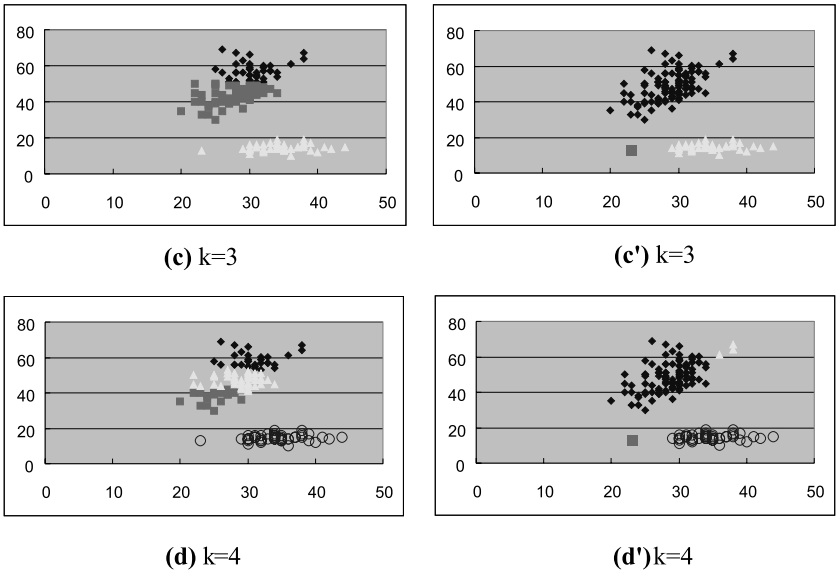

Fig. 2. Results of the second experiment. (c, d) are obtained by *k*-means algorithm while (c', d') are obtained by our outliers clustering process.

Table 2
Description of the 3 classes of sugar-cane breeding data

Cluster no.	DIA (cm)	LGH (m)	ESG (%)	FPC (%)
1	2.5–2.6	3.3–3.4	13.5–14.8	27–35
2	2.2–2.3	4.5–4.8	12.7–13.8	8–12
3	2.5–2.7	3.3–3.4	12.0–12.5	37–45

Table 3
Description of the 2 outliers of sugar-cane breeding data

Outlier no.	DIA (cm)	LGH (m)	ESG (%)	FPC (%)
1	2.4	4.2	7.8	50
2	3.2	5.8	11.2	47

behavior system, which works on the mail server simultaneously to analyze each mail received by the mail server. As we know, when user sends an E-mail through E-mail server, the server may record some information relative to user's behavior. Before we could feed the information from the log file into a relational database, we had to first clean and transform the raw data. The original log file entries include the following information: time, source and destination address, mail size, process priority, and user information, etc. An example is shown in Fig. 3.

Our experimental E-mail log data have 25511 entries, which are partitioned into six clusters. Based upon the experience of network administrators, we take two variables, count of E-mail entry (*X*-coordinate), size (log of bytes) of each E-mail entry (*Y*-coordinate), as our experimental parameters. Experimental results of clustering are shown in Fig. 4. From Fig. 4(a), we will find that the *k*-means algorithm just can divide these considerable normal patterns into several subclusters, but not differentiate outlier patterns. Fig. 4(b) shows that our cluster process can cluster more outlier patterns.

Nov	3	05:20:57	ccserv6	sendmail[26478]	OAA29303:	to=<tatami@flourix.com.tw>, ctladdr=<u8216012@ccserv6> (10421/16), delay=1+
Nov	3	05:20:57	ccserv6	sendmail[26478]	MAA28795:	to=<bbs@crystal.pine.ncu.edu.tw>, ctladdr=<u8543010@ccserv6> (42505/43), del
Nov	3	05:20:58	ccserv6	sendmail[26478]	TAA02509:	to=\void@void.dorm10.nctu.edu.tw, delay=1+09:52:37, xdelay=00:00:00, mailer=e
Nov	3	05:20:58	ccserv6	sendmail[26478]	LAA28364:	to=<l1685109@sparc1.cc.ncku.edu.tw>, ctladdr=<u8534524@ccserv6> (40344/34)
Nov	3	05:21:38	ccserv6	sendmail[26626]	FAA26626:	from=<owner-delphi-3d@deimos.frii.com>, size=1319, class=-60, pri=139319, nrc
Nov	3	05:21:38	ccserv6	sendmail[26627]	FAA26626:	to=<u8217816@cc.nctu.edu.tw>, delay=00:00:03, xdelay=00:00:00, mailer=local,
Nov	3	05:22:05	ccserv6	sendmail[26630]	FAA26630:	from=<u8618031@ccserv6>, size=458, class=0, pri=30458, nrcpts=1, msgid=<199
Nov	3	05:22:05	ccserv6	sendmail[26632]	FAA26630:	to=<u8618032@cc.edu.nctu.tw>, ctladdr=<u8618031@ccserv6> (45028/18), delay
Nov	3	05:22:06	ccserv6	sendmail[26632]	FAA26630:	FAA26632: DSN: Host unknown (Name server: cc.edu.nctu.tw: host not found)
Nov	3	05:22:06	ccserv6	sendmail[26632]	FAA26632:	to=<u8618031@ccserv6>, delay=00:00:00, xdelay=00:00:00, mailer=local, stat=S
Nov	3	05:22:06	ccserv6	sendmail[26632]	FAA26630:	FAB26632: postmaster notify: Host unknown (Name server: cc.edu.nctu.tw: host n
Nov	3	05:22:07	ccserv6	sendmail[26632]	FAB26632:	to=root, delay=00:00:01, xdelay=00:00:01, mailer=local, stat=Sent
Nov	3	05:24:29	ccserv6	sendmail[26636]	FAA26636:	from=<phcheng@MIT.EDU>, size=1703, class=0, pri=31703, nrcpts=1, msgid=<3

Fig. 3. An example of E-mail log file.

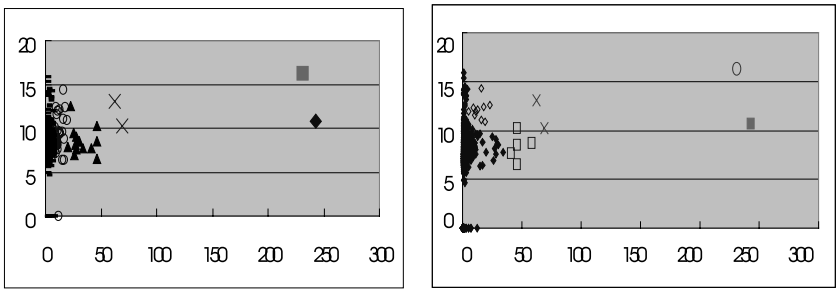

Fig. 4. Experiment results of E-mail log data file: (a) clustering by *k*-means algorithm, (b) clustering by our clustering method.

5. Concluding remarks

In traditional clustering algorithm, the anomaly patterns are either neglected or incorporated by other larger patterns. However, in some specific application, we have to find out these anomaly patterns from a large amount of data. In this paper, we proposed a two-phase clustering process to identify outliers. In first phase, we modified *k*-means algorithm based on a "splitting" heuristic; in the second phase, we proposed an OFP to find outliers from the resulting clusters obtained in Phase 1. In this phase, an MST is constructed and the longest edge is removed. Then replace the original tree with two newly generated subtrees. The small clusters, the tree with less number of nodes, are selected and regarded as outlier. Furthermore, we may iteratively remove the longest edges until the number of trees in the forest was sufficient enough. According to the first experiment: Iris data, the results show that our proposed method would find out the minor and anomaly patterns. In E-mail log file analysis experiment, the proposed clustering process is applied to an on-line monitoring of anomaly E-mail behavior system. The results show that our process works well.

Acknowledgements

This work was supported by Ministry of Education and National Science Council of the Republic of China under Grand No. 89-E-FA04-1-4, High Confidence Information Systems.

References

Ball, G.H., Hall, D.J., 1964. Some fundamental concepts and synthesis procedures for pattern recognition preprocessors. In: Proc. Internat. Conf. on Microwaves, Circuit Theory, and Information Theory, September, Tokyo.

Cedno, A.A., Suer, G.A., 1997. The use of a similarity codefficient-based method to perform clustering analysis to a large set of data with dissimilar parts. Comput. Industry Eng. 33, 225–228.

Davies, D.L., Bouldin, D.W., 1979. A clustering separation measure. IEEE Trans. Pattern Anal. Machine Intell. 1, 224–227.

Dubes, R.C., Jain, A.K., 1987. Algorithms that Cluster Data. Prentice-Hall, Englewood Cliffs, NJ.

Duda, R.O., Hart, P.E., 1973. Pattern Classification and Scene Analysis. Wiley Interscience, New York.

Forgy, E., 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biomertrics 21, 768.

Hadi, A.S., Simonoff, J.S., 1993. Procedures for the identification of multiple outliers in linear models. J. Amer. Statist. Assoc. 88, 1264–1272.

Hyvarinen, L., 1963. Classification of qualitative data. BIT 2, 83–88.

Jain, A.K., Dubes, R.C., 1988. Algorithms for Clustering Data. Prentice-Hall, NJ.

Jiang, M.F., Wang, C.H., Tseng, S.S., 1996. Developing a sugar-cane breeding assistant system by a hybrid adaptive learning technique. In: Proc. Internat. Conf. on Systems, Man, and Cybernetics, Beijing, China.

Johnson, R.A., Wichern, D.W., 1982. Applied Multivariate Statistical Analysis, 3rd ed. Prentice-Hall, Englewood Cliffs, NJ.

Kaufman, L., Rousseeuw, P., 1990. Finding Groups in Data: an Introduction to Cluster Analysis. Wiley, New York.

Kianifard, F., Swallow, W., 1990. A Monte Carlo comparison of five procedures for identifying outliers in linear regression. Comm. Statist., Part A, 1913–1938.

McQueen, J.B., 1967. Some methods of classification and analysis of multivariate observations. In: Proc. 5th Berkeley Symp. on Math. Statist. Probab., pp. 281–197.

Su, C.M., Tseng, S.S., Jiang, M.F., Chen, J.C.S., 1999. A fast clustering process for outliers and remainder clusters. Lecture Notes in Artificial Intell. 1574, 360–364.

Tou, J., Gonzalez, R., 1974. Pattern Recognition Principles. Addison-Wesley, Reading, MA.

Zahn, C.T., 1971. Graph-theoretical methods for detecting and describing Gestalt clusters. IEEE Trans. Comput. C 20, 68–86.