International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:9, 2008

K-Means for Spherical Clusters with Large Variance in Sizes

A. M. Fahim, G. Saake, A. M. Salem, F. A. Torkey, and M. A. Ramadan

***Abstract*—**Data clustering is an important data exploration technique with many applications in data mining. The k-means algorithm is well known for its efficiency in clustering large data sets. However, this algorithm is suitable for spherical shaped clusters of similar sizes and densities. The quality of the resulting clusters decreases when the data set contains spherical shaped with large variance in sizes. In this paper, we introduce a competent procedure to overcome this problem. The proposed method is based on shifting the center of the large cluster toward the small cluster, and re-computing the membership of small cluster points, the experimental results reveal that the proposed algorithm produces satisfactory results.

***Keywords*—**K-Means, Data Clustering, Cluster Analysis.

I. Introduction

THE huge amount of data collected and stored in databases increases the need for effective analysis methods to use the information contained implicitly there. One of the primary data analysis tasks is cluster analysis, which helps the user to understand the natural grouping or structure in a dataset. Therefore, the development of improved clustering algorithms has been received much attention. The goal of a clustering algorithm is to group the objects of a database into a set of meaningful subclasses [3].

Clustering is the process of partitioning or grouping a given set of patterns into disjoint clusters. This is done such that patterns in the same cluster are alike, and patterns belonging to two different clusters are different. Clustering has been a widely studied problem in a variety of application domains including data mining and knowledge discovery [10], data compression and vector quantization [11], pattern recognition and pattern classification [7], neural networks, artificial intelligence, and statistics.

Existing clustering algorithms can be broadly classified into hierarchical and partitioning clustering algorithms [17]. Hierarchical algorithms decompose a database D of n objects into several levels of nested partitioning (clustering), represented by a dendrogram, i.e., a tree that iteratively splits D into smaller subsets until each subset consists of only one object. There are two types of hierarchical algorithms; an agglomerative that builds the tree from the leaf nodes up, whereas a divisive builds the tree from the top down. Partitioning algorithms construct a single partition of a database D of n objects into a set of k clusters.

Optimization based partitioning algorithms typically represent clusters by a prototype. Objects are assigned to the cluster represented by the most similar prototype. An iterative control strategy is used to optimize the whole clustering such that, the average squared distances of objects to its prototypes are minimized. These clustering algorithms are effective in determining a good clustering, if the clusters are of convex shape, similar size and density, and if their number k can be reasonably estimated. Depending on the kind of prototypes, one can distinguish k-means, k-modes and k-medoids algorithms. In k-means algorithm [8], the prototype, called the center; is the mean value of all objects belonging to a cluster. The k-modes algorithm [16] extends the k-means paradigm to categorical domains. For k-medoids algorithms [7], the prototype, called the “medoid”; is the most centrally located object in the cluster. The algorithm CLARANS, introduced in [20], is an improved k-medoids type algorithm restricting the huge search space by using two additional user-supplied parameters. It is significantly more efficient than the well-known k-medoids algorithms PAM and CLARA, presented in [7].

Among clustering formulations that are based on minimizing a formal objective function, perhaps the most widely used and studied is k-means clustering. Given a set of n data points in real d-dimensional space, R^d, and an integer k, the problem is to determine a set of k points in R^d, called centers, so as to minimize the mean squared distance from each data point to its nearest center. Although the k-means method has a number of advantages over other data clustering techniques, it also has drawbacks; it converges often at a local optimum [2], the final result depends on the initial starting centers. Many researchers introduce some methods to select good initial starting centers; you can see [5] and [6]. Other researchers try to find the best value for the parameter k that determines the number of clusters or the value of k must be supplied by the user. You can see [22] and [21]. In recent years, many improvements have been proposed and

A. M. Fahim is with the faculty of information, Otto-von-Guericke University, Magdeburg, Germany (e-mail: ahmed.fahim@iti.cs.uni-magdeburg.de).

G. Saake is with the faculty of information, Otto-von-Guericke University, Magdeburg,Germany (e-mail: saake@iti.cs.uni-magdeburg.de).

A. M. Salem is with the faculty of Computers and Information, Ain Shams University, Cairo, Egypt (e-mail:absalem@asunet.shams.edu.eg).

F. A. Torkey is with the Kafer el Shiekh University, Kafer el Sheikh, Egypt (e-mail:fatorkey@Yahoo.com).

M. A. Ramadan is with the Faculty of Science, Minufiya University, Shbien el koum, Egypt (e-mail: mramadan@mailer.eun.eg).

implemented in the K-means method; you can see [9]. The k-means clustering algorithm attempts to determine k partitions that optimize a certain criterion function. The average square-error criterion, defined in (1), is the most commonly used (m_i is the mean of cluster C_i, n is the number of objects in the dataset).

$$E = \frac{1}{n}\sum_{i=1}^{k}\sum_{x\in c_i}(x-m_i)^2 \quad (1)$$

The average square-error is a good measure of the within-cluster variation across all the partitions. Thus, the average square error clustering tries to make the k clusters as compact and separated as possible, and works well when clusters are compact clouds that are rather well separated from one another [12]. However, when there are large differences in the sizes or geometries of different clusters, as illustrated in Figure 1, the square-error method could split large clusters to minimize the square-error. The square-error is larger for the three separate clusters in left than for the three clusters in right, where the big cluster is split into three portions, one of which is merged with one of the two smaller clusters. The reduction in square-error (in right) is due to the fact that the slight reduction in square error due to splitting the large cluster is weighted by many data points in the large cluster. We propose a competent idea to solve this problem.

There are very large number of clustering algorithms appeared [24], [12], [8], [23], [15], [1] and [25] but, in this paper, we focus on the k-means algorithm, a new procedure is added to the k-means algorithm makes it able to discover clusters with large variance in sizes with small separation between clusters. So we will review the k-means and some variants of it in section II, discuss the proposed idea in section III, present some experimental results in section IV and conclude with section V.

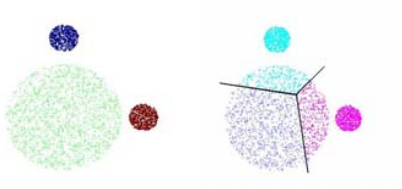

Fig. 1 Splitting of a large cluster by k-means algorithm.

II. Related Work

The k-means algorithm uses the mean value of the objects in a cluster as the cluster center. Suppose that a dataset of n objects $x_1, x_2, ..., x_n$ such that each object is in R^d, the problem of finding the minimum variance clustering of the dataset into k clusters is that of finding k points m_i, $I = 1, 2, ..., k$, in R^d such that Equation (1) is minimized. The basic processes of the k-means algorithm are:

1. Initialization: Select a set of k starting points m_j, $j = 1, 2, ..., k$. The selection may be done in random manner or according to some heuristic.
2. Distance calculation: For each object x_i, $1 \leq I \leq n$ compute its Euclidean distance to each cluster centroid m_j, $1 \leq j \leq k$, and then find the closest cluster centroid.
3. Centroid recalculation: For each $1 \leq j \leq k$ ecomputed cluster centroid m_j as the average of the data points assigned to it.
4. Convergence condition: Repeat step 2 and 3 until convergence.

Before the k-means algorithm converges, step2 and step3 are executed number of times, say j, where the positive integer j is known as the number of k-means iterations. The precise value of j varies depending on the initial starting clusters centroids even on the same data set. So the computational time complexity of the algorithm is O(nkj), Where n is the total number of objects in the dataset, k is the required number of clusters we identified and j is the number of iterations, $k \leq n$, $j \leq n$. The k-means algorithm can be thought of as a gradient descent procedure which begins at the starting clusters centroids and iteratively updates these centroids to minimize the objective function in equation (1). It is known [4] that, the k-means will always converge to a local minimum. When we analyze the k-means we find that, the main advantages of this algorithm are; (1) its efficiency, (2) this algorithm is very easy to implement and (3) speed of convergence. On the other hand, its main drawbacks are (1) the final result depends on the initial starting centers, (2) to choose a proper number of clusters k is a domain dependent problem, (3) this algorithm is applicable only when mean is defined, (4) it is sensitive to outliers and (5) this algorithm is Good only for convex shaped, similar size and density clusters. For the first four disadvantages, there are a lot of efforts have been done to overcome these problems, we review some of them, the proposed method handles the last problem.

Several variants of the k-means algorithm have been proposed. Their purpose is to improve efficiency or find better clusters; improved efficiency is usually accomplished by either reducing the number of iterations to reach final convergence or reducing the total number of distance calculations. Therefore, choosing a good set of initial cluster centers is very important for the algorithm. However, it is difficult to select a good set of initial cluster centers randomly. Bradley and Fayyad [5] have proposed an algorithm for refining the initial cluster centers. The main idea of their algorithm is to select m subsamples from the data set, apply the k-means on each subsample independently, keep the final k centers from each subsample provided that empty clusters are not be allowed, so they obtain a set contains mk points. They apply the k-means on this set m times; at the first time, the first k points are the initial centers. At the second time, the second k points are the initial centers, and so on. And the algorithm returns the best k centers from this set. They use 10 subsamples from the data set, each of size 1% of the full dataset size.

To choose a proper number of clusters k is a domain dependent problem. To resolve this problem, some methods

have been proposed to perform k-clustering for various numbers of clusters and employ certain criteria for selecting the most suitable value of *k* [21] and [22]. For example in [22] the authors depend on the fact that, the k-means method aims to minimize the sum of squared distances from all points to their cluster centers, this should result in compact clusters. So they use the distances of the points from their cluster centers to determine whether the clusters are compact or not. For this purpose, they use the intra-cluster distance measure, which is simply the distance between a point and its cluster center and take the average of all of these distances, as defined in equation (1). So the intra-cluster distance is the average squared error that the k-means method minimizes. Also the authors measure the inter-cluster distance, or the distance between clusters, which should to be as large as possible. So they calculate this as the distance between cluster centers, and take the minimum of these values, defined as in equation (2)

$$\text{int}\,er = \min(\|m_i - m_j\|^2),$$
$$i = 1, 2, ..., k-1 \text{ and } \; j = i+1, ..., k \qquad (2)$$

They take only the minimum of these values as they want the smallest of this distance to be maximized, use these measures to help them to determine if they have a good clustering, so they minimize the ratio between them, defined as in equation (3).

$$validity = \frac{\text{int}\,ra}{\text{int}\,er} \qquad (3)$$

Therefore, the clustering which gives a minimum value for the validity measure will give the ideal value of *k* in the k-means algorithm. The k-means algorithm is applicable only when mean is defined, also this problem is solved by introducing the k-modes algorithm [16]. This is an extended version of the k-means with some modification to be suitable for categorical data. The cause that the k-means algorithm can not cluster categorical objects is its dissimilarity measure and the method used to solve the clustering problem. These barriers have been removed by making the following modifications to the k-means algorithm

1. Using a simple matching dissimilarity measure for categorical objects.
2. Replacing means of clusters by modes.
3. Using a frequency-based method to find the modes to solve the problem.

It is known that the k-means is sensitive to outliers, but there are some researches have been done to solve this problem. Some of them are to detect the outliers [14] first and remove them, and then apply a clustering algorithm. In [13] an Outlier Removal Clustering (ORC) algorithm is proposed, that provides outlier detection and data clustering simultaneously. This algorithm consists of two consecutive stages, which are repeated several times. In the first stage, they perform K-means algorithm with multiple initial starting points, and pick the best centers, and in the second stage, the algorithm assigns an outlyingness factor for each point and iteratively removes the points which are far from their clusters centers. This factor depends on the point's distance from the cluster center and the most far point from the cluster center. And all points with factor greater than specified threshold are considered as outliers and removed from the data set. The k-means algorithm is good only for convex shaped, similar sizes and density clusters. We propose a competent method in the following section to overcome this problem.

III. Our Algorithm (Refinement of the Final Clusters)

The k-means algorithm is a popular clustering algorithm and has its application in data mining, image segmentation [22], bioinformatics and many other fields. This algorithm works well with spherical shaped clusters of similar sizes. In this section we present how to make this algorithm works well with spherical shaped clusters of any size. In our proposed method, we find the distances between the means result from the k-means algorithm. For each cluster, we calculate its average radius of a cluster C_i -by dividing the sum of squared error of its points from its representative by the number of points assigned to it- as in equation (4).

$$radius(c_i) = \frac{\sum_{p \in c_i} d^2(p, m_i)}{n_i} \qquad (4)$$

We search for the largest cluster (have the largest average radius) and test whether this cluster have some portion merged with other clusters or not. At the first time, you can say if the summation of the two radiuses is larger than the distance between the two clusters, then there is a portion of the large cluster merged with the other cluster. So we can redistribute the points in the smaller cluster only over the two clusters. But this formula is not suitable at all. Since the mean of cluster is the center of gravity of the points. Also the average radius (as in equation 4) for both clusters is larger than the actual radius, and the small cluster attracts portion of points from the large cluster, and this leads to enlargement of the actual radius of the small cluster, and the mean of the smaller cluster is attracted toward the larger cluster, since the objective of the k-means is to get the smallest value for the squared error function in equation (1). So, we use equation (6) to get the sum of the two radiuses and compare this value with the distance between the two clusters which calculated as in equation (5). Where m_L and m_s are the means of the large and the small cluster respectively, *d* is the dimensionality of the data, *L* and *S* refer to the large and small cluster respectively.

$$Meandis\tan ce = \sqrt{\sum_{i=1}^{d} (m_{li} - m_{si})^2} \qquad (5)$$

$$somofradius = (radius(L) + radius(S)) * 0.80 \qquad (6)$$

If the *Sumofradius* is larger than or equal to the *Meandistance* and at the same time the ratio between the two radius is smaller than 0.90, -this condition is used to insure

that there is large difference in size-, then some portion of the larger cluster is merged with the smaller one. In this case we must redistribute the points in the smaller cluster to return the misclassified points to the larger cluster. How can we redistribute the points in the smaller cluster? To do this operation, we take the average of the two means as new mean ($Avmean = \frac{m_l + m_s}{2}$) . This new mean is located in the large cluster, and at the mid distance between the two cluster centers. We redistribute the points in the small cluster over the new mean (*Avmean*) and it's original mean (m_s). Since the means of the two clusters are shifted. So, we use the next formula to redistribute the points of the smaller cluster. We add a small value to the right hand side of relation 7, since we expect a small separation between the two clusters and the mean of the smaller cluster is attracted to the larger cluster. We multiply the ratio between the two radiuses by 0.80 since the radius of the smaller cluster has error percent larger than the other cluster

$$Dis(pi, Avmean) \le Dis(p_i, m_s) + \frac{radius(L) * 0.8}{radius(s)} \quad (7)$$

If the formula 7 is true then the point p_i is moved to the larger cluster, otherwise it remains in the smaller cluster. After redistribution of the all points in the smaller cluster, we recalculate the new means for the two clusters. All these processes are repeated for all clusters. So the final means of the proposed method are better than those produced first from the k-means algorithm. Fig.2 shows the proposed function added to the k-means algorithm to improve the final results.

```
For i=1 to k
   For j=i+1 to k
      Compute Euclidean distance d (m_i, m_j);
   Next j
 Next i
For j=1 to k/2
 Find the largest cluster L
 For i=1 to k
  If (radius(L)> radius(I)
    S=I
       Sum_of_radius = (radius(L) + radius(S) )*0.80
      If (Sum_of_radius >= d (m_L, m_S)&&( radius(S) / radius(L)) < 0.90)
        Av_mean=(m_L+m_S)/2
        Redistribute points represented by m_S over the two means m_S and Av_mean.
        All points assigned to Av_mean are moved to the cluster m_L.
         Find the new mean for the cluster m_S.
      Endif
    Endif
 Next i
 Find the new mean for the cluster m_L.
Next j
```

Fig. 2 Refinement process of the final results of the k-means algorithm

IV. Experimental Results

In this section, we present some experimental evaluation of the proposed algorithm, which reveal a great improvement in the k-means algorithm when the dataset contains spherical shaped clusters with large variance in their sizes. We have created many 2-dimensional datasets that contain spherical clusters of large different size. These datasets are created according to the general equation of circle $(x-a)^2 + (y-b)^2 \le r^2$. We present here some of them. All these datasets contain three clusters as shown in Figures from 3 to 6. Table I presents the exact number of points in each cluster. Also we present the exact number of points in each cluster founded by the k-means and by our proposed algorithm.

The experimental results in Table I which sketched by figures from 3 to 7 show the great improvement at the final clusters discovered by the proposed algorithm. When we examine the percent of error in the proposed algorithm, we find that, the proposed method produces the exact clusters in dataset 1 and dataset 2, because there is a separation between clusters. But there is a small error at clusters discovered from dataset 3, note that, there is no separation between clusters. From Table I you can see that, there are 7 points misclassified.

TABLE I
Comparison between the Results from the K-Means and our Proposed Method

Data set	Exact clusters	k-mean clusters	k-means error	Proposed clusters
Set1	1815	1210	605 points	1815
	683	973		683
	660	975		660
Set2	1582	1025	557 points	1582
	703	1015		703
	642	887		642
Set3	1582	1043	539 points	1585
	557	816		552
	522	802		524
Set4	2129	1306	823 points	2067
	505	916		534
	510	922		543
Set5	2363	1417	946 points	1417
	522	992		992
	524	1000		1000

Fig. 5 shows that our proposed algorithm not produces the exact clusters, this is occurred because the two centers of the two small clusters lie on the border of them, they are very close to the large cluster. But the result is better than the original result from the k-means. So as the large cluster become more densely the quality of our proposed algorithm degenerates, because the high density of points in large cluster attract the centers of the other smaller clusters, and the situation become more complex when the center of the smaller cluster belong to the larger one as in Fig. 6 which shows that both algorithms produce the same result, in other words, as the density of the larger cluster increase and the density difference

between the two clusters increase and the density of smaller cluster decrease, both algorithms produce the same result. Fig. 7 summaries our experimental results.

Fig. 3 Resulting Clusters from the k-means algorithm (datasets 1, 2 and 3)

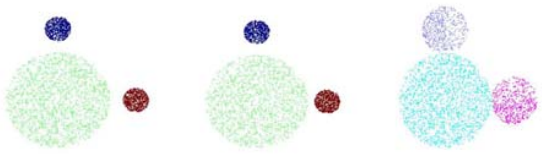

Fig. 4 Resulting Clusters from the proposed method (datasets 1, 2 and 3)

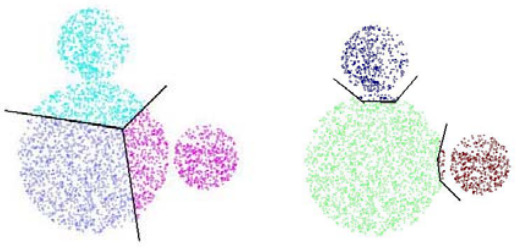

Fig. 5 k-means Result (on left), proposed method's result (on right) (dataset 4)

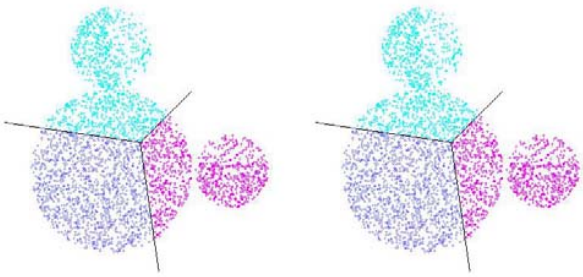

Fig. 6 Both algorithms produce the same result, because the three centers lie on the large cluster. and the difference between the average radius is very small; i.e the clusters seems to have equal radius (dataset 5)

A. Time Complexity

As we know that, the time complexity of the k-means algorithm is O(*nkj*); where *n* is the number of data points in the dataset, *k* is the number of clusters and *j* is the number of iterations. Since we use the k-means and then we apply our procedure, so the time complexity is equal to the summation of the two times. At first, our method find the distances between the pair wise *k* clusters so this operation requires $O(k^2)$. Then we search for the largest cluster, that requires O(*k*), at most the points of 3 or 4 clusters will be redistributed over their means and the average means, this operation requires O(2*mh*); 2 is the two means, *m* is the number of points in the cluster and *h* is the number of clusters we redistribute their points. Since we redistribute the points in the smaller cluster so $m < \frac{n}{2k}$, and *h* is very small, we can say *h* = 4 at most. So the time complexity added to the k-means is very small compared with the time complexity of the k-means itself. So, the time is $O(k^2 + mh)$, $k<n$. At the end the time complexity is $O(nkj + k^2 + mh)$.

V. Conclusion

In this paper, we have described a new procedure added to the end of the k-means clustering algorithm. The objective of this procedure is to refine the results of the k-means. This procedure is optional and it is strongly recommended to use it after the k-means especially when the dataset contains spherical shaped clusters with large difference in their sizes. Our experimental results are evidence that our proposed method improve the quality of the resulting clusters. our proposed algorithm produce the same result as k-means when the centers of the smaller clusters lie out of them, because in this situation the clusters seem to have very small difference between their radius. In future work we will search for more robust solution for this problem.

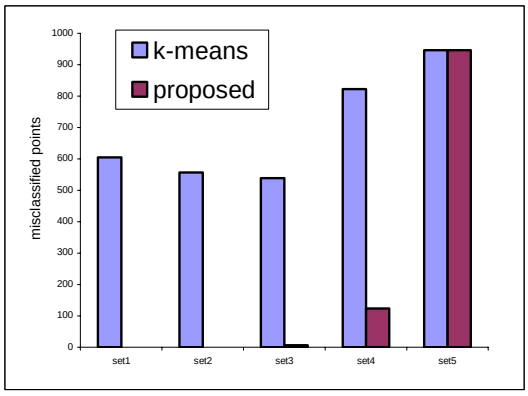

Fig. 7 The efficiency of the proposed method

References

[1] R. Agrawal , J. Gehrke , D. Gunopulos, and P. Raghavan, "Automatic SubspaceClustering of High Dimensional Data for Data Mining Applications," Proc.ACM SIGMOD Int. Conf. on Management of Data, Seattle, WA, 1998, pp. 94-105.

[2] M R. Anderberg *Cluster Analysis for Applications*, Academic Press, 1973.

[3] M. Ankerst , M. Breunig , H.P. Kriegel , and J. Sander, "OPTICS: Ordering Points to Identify the Clustering Structure," Proc. ACM SIGMOD Int. Con. Management of Data Mining, 1999, pp. 49-60.
[4] L. Bottou, and Y. Bengio, "Convergence properties of the k-mean algorithm," The MIT press, Cambridge, MA, 1995, pp. 585-592.
[5] P. S. Bradley , and U. M. Fayyad, "Refining Initial Points for K-Means Clustering," Proc. of the 15th International Conference on Machine Learning (ICML98), J. Shavlik (ed.). Morgan Kaufmann, San Francisco, 1998, pp. 91-99.
[6] S. Deelers, and S. Auwatanamongkol, "Enhancing K-Means Algorithm with Initial Cluster Centers Derived from Data Partitioning along the Data Axis with the Highest Variance," PWASET vol 26, 2007, pp. 323-328.
[7] R.O. Duda , and P.E. Hart, *Pattern Classification and Scene Analysis*. John Wiley Sons, New York, 1973.
[8] M. Ester, H.P. Kriegel, J. Sander, and X. Xu, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, AAAI Press, Portland, 1996, pp.226-231.
[9] A. M. Fahim, A. M. Salem ,F. A. Torkey, and M. Ramadan, "An efficient enhanced k-means clustering algorithm," Journal of Zhejiang University SCIENCE A, vol 7(10), 2006, pp. 1626-1633.
[10] U.M. Fayyad , G. Piatetsky-Shapiro , P. Smyth , and R. Uthurusamy *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, 1996.
[11] A. Gersho, and R.M. Gray *Vector Quantization and Signal Compression*, Kluwer Academic, Boston, 1992.
[12] S. Guha , R. Rastogi, and K. Shim, "CURE: An Efficient Clustering Algorithms for Large Databases," Proc. ACM SIGMOD Int. Conf. on Management of Data, Seattle, WA, 1998, pp. 73-84.
[13] V. Hautamaeki , S. Cherednichenko , I. Kaerkkaeinen , T. Kinnunen, and P. Fraenti, "Improving K-Means by Outlier Removal," SCIA 2005, LNCS 3540, 2005, pp. 978-987.
[14] V. Hautamaeki , I. Kaerkkaeinen, and P. Fraenti, "Outlier detection using k-nearest neighbourgraph," In: 17th International Conference on Pattern Recognition (ICPR 2004), Cambridge, United Kingdom, 2004, pp. 430-433.
[15] A. Hinneburg, and D. Keim, "An Efficient Approach to Clustering in Large Multimedia Databases with Noise," Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining, New York City, NY. 1998.
[16] Z. Huang, "A fast clustering algorithm to cluster very large categorical data sets in data mining," Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Dept. of Computer Science, The University of British Columbia, Canada, 1997, pp. 1-8.
[17] A. K. Jain, and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.
[18] L. Kaufman, and P. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis," Wiley, 1990.
[19] J.B. MacQueen, "Some methods for classification and analysis of multivariate observations. Proc. 5th Symp. Mathematical Statistics and Probability, Berkelely, CA, Vol(1), 1967, pp. 281297.
[20] R.T. Ng, and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," Proc. 20th Int. Conf. on Very Large Data Bases. Morgan Kaufmann Publishers, San Francisco, CA, 1994, pp. 144-155.
[21] D. T. Pham , S. S. Dimov, and C. D. Nguyen, "Selection of *k* in K-means clustering," Mechanical Engineering Science, vol(219), 2004, pp. 103-119.
[22] S. Ray, and R. H. Turi, "Determination of number of clusters in k-means clustering and application in colour image segmentation," In Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques, 1999, pp.137-143.
[23] G. Sheikholeslami, S. Chatterjee, and A. Zhang, "Wave-Cluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases," Proc. 24th Int. Conf. on Very Large Data Bases. New York, 1998, pp. 428-439.
[24] R. Sibson, "SLINK: an optimally efficient algorithm for the single-link cluster method," The Comp. Journal, 16(1), 1973, pp. 30-34.
[25] T. Zhang, R. Ramakrishnan, and M. Linvy, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," Proc. ACM SIGMOD Int. Conf. on Management of Data. ACM Press, New York, 1996, pp. 103-114.