CLUSTERING TECHNIQUE IN DATA MINING : GENERAL AND RESEARCH PERSPECTIVE

Zamzarina Che Mat @ Mohd Shukor, Mohd Noor Md Sap
Faculty of Computer Science and Information System
University Technology of Malaysia, K.B. 791
81310 Skudai, Johor, Malaysia
Tel : (607)-5576160, Fax : (607) 5566155
Emails: mohdnoor@fsksm.utm.my, azmiraz@hotmail.com,

ABSTRACT

As the amount and dimensionality of data grows beyond the grasp of human minds, automation of pattern discovery becomes crucial. One of the most popular techniques to extract pattern and knowledge from large amount of data in databases is data mining. Data mining can be defined as process of searching the particular patterns and relationship from large amount of data in databases using sophisticated data analysis tools and techniques to build models that may be used to make valid predictions. One of the existing data mining techniques is clustering. Clustering in data mining is a discovery process that groups a set of data such that the intra-cluster similarity is maximized and inter-cluster similarity is minimizes. These discovered clusters are used to explain the characteristics of the data distribution. This paper present most popular clustering technique such as hierarchical clustering and partitional clustering, cluster selection schemes, clustering criterion functions, assessing cluster quality and conclusion.

Keywords : Clustering, Data Mining, Unsupervised Learning, Descriptive Learning, Partitioning, Hierarchical Clustering, Agglomerative.

1 INTRODUCTION

With the rapid growth in number of available databases, mining knowledge from database becomes essential to predict future behavior and support decision-making. There are many available tools can be used to extracted numerous valuable knowledge from databases such as QBE (Query By Example) and SQL (Structure Query Language). However these techniques cannot extract the knowledge from databases and can only extract the meaning of data. One of the most popular tools to extract meaning and knowledge from database is data mining.

Data mining can be defined as process of searching the particular patterns and relationship from large amount of data in databases using sophisticated data analysis tools and techniques to build models that may be used to make valid predictions. Jun and Keng (1999) defines data mining as searching for valuable information in large volumes of data. It is the essential process of extraction of implicit previously unknown and potentially useful information such as knowledge, rules, constraints and regularities from data stored in repositories using pattern recognition technologies. Data mining techniques can be classified into the following categories: classification, clustering, association rules, sequential patterns, time series patterns, link analysis and text mining.

Clustering is an exploratory tool for analyzing large datasets, and has been used extensively in numerous application areas. It is the task of organizing a set of objects into

meaningful groups. These groups can be disjoint, overlapping, or organized in some hierarchical fashion. The key element of clustering is the notion that the discovered groups are meaningful (Zhao and Karypis, 2002). Clustering techniques can be classified into two main techniques: hierarchical and partitional technique. Stone et. al,.(1993) defined clustering in data mining as a discovery process that groups a set of data such that the intra-cluster similarity is maximized and the inter-cluster similarity is minimized. This discovered clusters used to explained the characteristics of underlying data distribution. In some applications, even through their overall similarity is not the highest, it also can be translate to groups that contain objects that share some key characteristics. Clustering is put in historical perspective by data modeling that rooted in mathematics, statistics, and numerical analysis. From machine learning perspective, clusters correspond to hidden patterns, the search of clusters is unsupervised learning, and the resulting system represents a data concept. Therefore, clustering is unsupervised learning of a hidden data concept (Berkhin et. al, 2002).

To facilitate the readers, explanation of this paper is organized into some sections listed as follows. The descriptive of Hierarchical Techniques, Partitioning Techniques and Other Clustering Techniques is given in section 2, and 3. Section 4 describes the cluster selection schemes and it is continued with the explanation of clustering criterion functions in section 5. Finally section 6 gives the conclusion of clustering technique in research perspective.

2 HIERARCHICAL TECHNIQUE

George Karypis et. al, (1999) defined hierarchical clustering as the algorithms that produce a nested sequence of clusters, with a single all-inclusive cluster at the top and single point clusters at the bottom. Hierarchical clustering builds a cluster hierarchy also known as a *dendrogram*. This tree of clusters contains cluster node that have child clusters; sibling clusters partition the points covered by their parent. Such an approach allows exploring data on different levels of granularity. Kaufman and Rousseeuw, (1990) categorized hierarchical clustering methods into *agglomerative* (bottom-up) and *divisive* (top-down). An *agglomerative* clustering starts with singleton clusters and recursively merges two or more most appropriate clusters. A *divisive* clustering starts with one cluster of all data points and recursively splits the most appropriate cluster. The process continues until a stopping criterion is achieved. Advantages of hierarchical clustering include:

i. Embedded flexibility regarding the level of granularity
ii. Ease of handling of any forms of similarity or distance
iii. Consequently, applicability to any attribute types

Disadvantages of hierarchical clustering are related to:

i. Vagueness of termination criteria
ii. The fact that most hierarchical algorithms do not revisit once constructed (intermediate) clusters with the purpose of their improvement

The traditional approaches to hierarchical clustering are presented in the sub-section *Linkage Metrics*. Hierarchical clustering based on linkage metrics results in clusters of proper shapes. Active contemporary efforts to build cluster systems that incorporate intuitive concept of clusters as connected components of arbitrary shape, are surveyed in the sub-section *Hierarchical Clusters of Arbitrary Shapes*.

2.1 Linkage Metrics

Hierarchical clustering initializes a cluster system as a set of singleton clusters (agglomerative case) or a single cluster of all points (divisive case). It proceeds iteratively with merging or splitting of the most appropriate clusters until the stopping criterion is achieved. The appropriateness of clusters for merging/splitting depends on the similarity/dissimilarity of clusters elements. This reflects a general presumption that clusters consist of similar points. Example of dissimilarity between two points is the distance between them.

To merge or split subsets of points rather than individual points, the distance between individual points has to be generalized to the distance between subsets. Such derived proximity measure is called a *linkage metric*. The type of the linkage metric used significantly affects hierarchical algorithms, since it reflects the particular concept of *closeness* and *connectivity*. Major inter-cluster linkage metrics (Olson 1995) include *single link*, *average link*, and *complete link*. The underlying dissimilarity measure (usually, distance) is computed for every pair of points with one point in the first set and another point in the second set. A specific operation such as single link, average link, or complete link is applied to pair-wise dissimilarity measures:

$$d(C_1, C_2) = operation \{ d(x, y) \mid x \in C_1, y \in C_2 \}. \quad (1)$$

where C is the simplest attribute space subset which is a direct Cartesian product of subranges called segment. These subranges consist of a single category value, or of a numerical bin.

Early examples include the algorithm SLINK (Sibson, 1973), which implements single link. Defays, (1977) proposed the algorithm CLINK, which implements complete link, and the average link is implements in Voorhees' method which proposed by Voorchees (1986). Of these SLINK is referenced the most and it is related to the problem of finding the Euclidean minimal spanning tree that has N sample, and has $O(N^2)$ complexity. The methods using inter-cluster distances defined in terms of pairs with points in two respective clusters (subsets) are called *graph* methods. When the connectivity N x N matrix is specified, graph methods directly dealing with the connectivity graph can be used. They do not use any cluster representation other than a set of points. This name naturally relates to the connectivity graph, since every data partition corresponds to a graph partition. Such methods can be appended by so-called *geometric* methods in which a cluster is represented by its central point. It results in *centroid*, *median*, and *minimum variance* linkage metrics. Under the assumption of numerical attributes, the center point is defined as a centroid. It also can be defined as an average of two cluster centroids subject to agglomeration. All of the above linkage metrics can be derived as instances of the Lance-Williams updating formula (Lance and Williams, 1967).

$$d(C_i \cup C_j, C_k) = a(i)d(C_i, C_k) + a(k)d(C_j, C_k) + bd(C_i, C_j) + c \mid d(C_i, C_k) - d(C_j, C_k) \mid . \quad (2)$$

Here a, b, c are coefficients corresponding to a particular linkage. This formula expresses a linkage metric between the union of the two clusters and the third cluster in terms of underlying components. The ultimate goal is to assign points of finite system of k subsets, clusters. The Lance-Williams formula has an utmost importance since it makes manipulation with similarity/dissimilarity computationally feasible. When the base measure is distance, these methods capture inter-cluster closeness. However, a similarity based view that results in intra-cluster connectivity considerations is also possible. This is how original average link agglomeration (Group-Average Method) (Jain and Dubes, 1988) was introduced.

Linkage metrics-based hierarchical clustering suffers from time complexity. Under reasonable assumptions, such as *reducibility condition* (graph methods satisfy this condition), linkage metrics methods have $O(N^2)$complexity (Olson, 1995). Despite the unfavorable time complexity, these algorithms are widely used. An example is algorithm AGNES proposed by Kaufman and Rousseeuw (1990) that used in S-Plus.

2.2 Hierarchical Clusters of Arbitrary Shapes

Linkage metrics based on Euclidean distance for hierarchical clustering of spatial data naturally predispose to clusters of proper convex shapes. Meanwhile, visual scanning of spatial images frequently attests clusters with curvy appearance. There are many variations of agglomerative hierarchical algorithms (A.K Jain and R.C. Dubes, 1988). In some methods, each cluster is represented by a centroid or medoid of the points contained in the cluster, and the similarity between two clusters is measured by the similarity between the centroids/medoids of the clusters. These methods fail on clusters of arbitrary shapes and different sizes. Agglomerative hierarchical algorithms find the clusters by initially assigning each object to its own cluster and then repeatedly merging pairs of clusters until a certain stopping criterion is met. Consider an n-object dataset and the clustering solution that has been computed after performing l merging steps. This solution will exactly $n - l$ clusters, as each merging step reduces the number of clusters by one. Given $(n - l)$-way clustering solution, the pair of clusters that selected to be merged next, is the one that leads to a $(n - l - 1)$-way solution that optimizes a particular criterion function. That is each one of the $(n - l)$ x $(n - l - 1)/2$ pairs of possible merges is evaluated, and the one that leads to a clustering solution that has the maximum (or minimum) value of the particular criterion function is selected. Thus the criterion function is locally optimized within each particular stage of agglomerative algorithms. Depending on the desired solution, this process continues until either there are only k clusters left, or when the entire agglomerative tree has been obtained.

The rationale for this approach is that sub clusters belonging to the same cluster will tend to have high interconnectivity. But the aggregate inter-connectivity between two clusters depends on the size of clusters involved, and in general pairs of larger clusters will have higher inter-connectivity. Hence, many such schemes normalize the aggregate similarity between a pair of clusters with respect to the expected inter-connectivity of the clusters involved. For example, the widely used group-average method assumes fully connected clusters, and thus scales the aggregate similarity between two clusters by n x m, where n and m are the size of the two clusters, respectively (A.K. Jain and R.C. Dubes, 1988).

Guha et al. (1998) introduced the hierarchical agglomerative clustering algorithm CURE. CURE has been proposed to remedy the drawbacks of both of these methods while combining their advantages (Kyuseok Shim et. al, 1998). A major feature of CURE is that it represents a cluster by a fixed number c of points scattered around it. In CURE, instead of using a single centroid (geometric) to represent a cluster, a constant number of representative points are chosen to represent a cluster. The similarity between two clusters is measured by the similarity of the closest pair of the representative points belonging to different clusters. New representative points for the merged clusters are determined by selecting a constant number of well scatter points from all the data points and shrinking them towards the centroid of the cluster according to a shrinking factor α. Unlike centroid or medoid based methods, CURE is capable of finding clusters of arbitrary shapes and sizes, as it represents each cluster via multiple representative points and it is insensitive to outliers. Shrinking the representative points towards the centroid helps CURE in avoiding the problem of noise and outliers present in the single link method. Since CURE uses sampling, estimation of its complexity is not straightforward. For

low-dimensional data authors provide a complexity estimate of $O(N^2)$ defined in terms of sample size. More exact bounds depend on input parameters: shrink factor α, number of representative points c, number of partitions, and sample size. Figure 1 illustrates agglomeration in Cure. Three clusters, each with three representatives, are shown before and after the merge and shrinkage. Two closest representatives are connected by arrow.

While the algorithm CURE works with numerical attributes (particularly low dimensional spatial data), the algorithm ROCK developed by the same researchers (Guha et al., 1999) targets hierarchical agglomerative clustering for categorical attributes that employs links and not distance for merging cluster. Compared to traditional clustering algorithm that used distance between points of clustering, ROCK appropriate concept of links to measure the similarity/proximity between a pair of data points with categorical attributes. ROCK, operates on a derived similarity graph, scales the aggregate inter-connectivity with respect to a user inter-connectivity model. This method is naturally extended to non-matrix similarity measures. ROCK generates better quality clusters than traditional algorithms and exhibits good properties. However, CURE and ROCK fail to take into account special characteristics of individual clusters. These schemes also can make incorrect merging decisions when the underlying data does not follow the assumed model, or when noise is present.

Next, the hierarchical agglomerative algorithm CHAMELEON proposed by Karypis et al. (1999) utilizes dynamic modeling in cluster aggregation. This algorithm measures the similarity of two clusters based on dynamic model. CHAMELEON has two stages. In the first stage small tight clusters are built to ignite the second stage. This involves a graph partitioning (Karypis and Kumar 1999). In the second stage agglomerative process is performed. It utilizes measures of relative inter-connectivity $RI(C_i, C_j)$ and relative closeness $RC(C_i, C_j)$; both are locally normalized by quantities related to clusters. The merging process using dynamic model that presented in CHAMELEON discovery of natural and homogeneous clusters. This algorithm used methodology of dynamic modeling of clusters that applicable to all types of data as long as similarity matrix can be constructed. The algorithm does not depend on assumptions about the data model. This algorithm is proven to find clusters of different shapes, densities, and sizes in 2D (two-dimensional) space. Past algorithm like CURE and ROCK can breakdown if the model is not adequate to capture the characteristics of clusters. Furthermore, most of these past algorithms breakdown when the data of clusters consists that are of divisive shapes, densities and sizes. CHAMELEON can discover natural clusters that many existing state-of-art of clustering algorithms fail to find.

Figure 2 (analogous to the one in (Karypis and Kumar, 1999) presents a choice of four clusters (a) - (d) for a merge. While Cure would merge clusters (a) and (b), CHAMELEON makes intuitively better choice of merging (c) and (d).

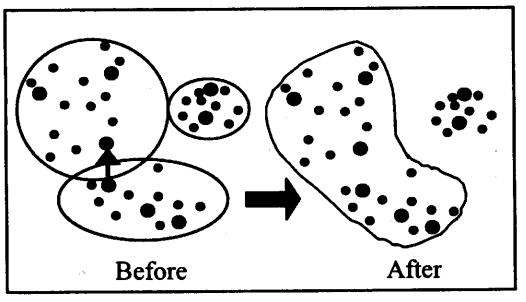


Figure 1: Agglomeration in Cure.

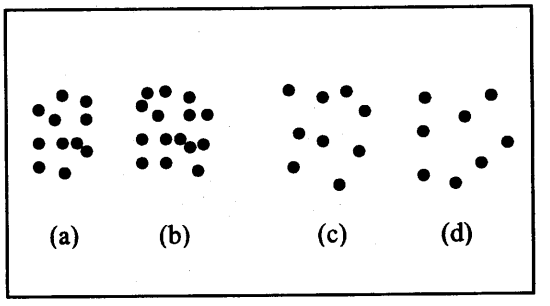


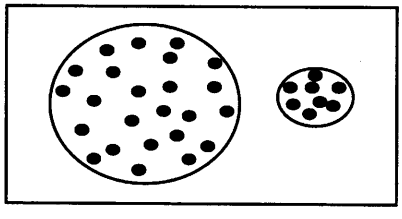
Figure 2: CHAMELEON merges (c) and (d)

3 PARTITIONAL TECHNIQUE

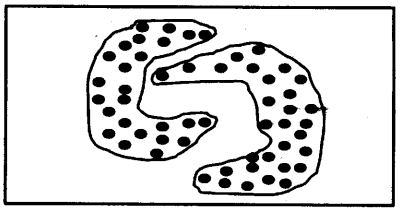
Partitional clustering attempts to break a data set into K clusters such that the partition optimizes a given criterion (A.K. Jain and R.C Dubes, 1988). Partitional clustering algorithms compute a *k*-way clustering of a set of objects either directly or via a sequence of repeated bisections. A direct *k*-way clustering is commonly computed as follow. Initially, a set of *k* objects is selected from the datasets to act as the seeds of the *k* clusters. Then, for each object, its similarity to these *k* seeds is computed, and it is assigned to the cluster corresponding to its most similar seeds. This forms the initial *k*-way clustering. This clustering is then repeatedly refined so that it optimizes a desired clustering criterion function. A *k*-way partitioning via repeated bisections is obtained by recursively applying the above algorithm to compute 2-way clustering (i.e., bisections). Initially, the objects are partitioned into two clusters, then the one of these clusters is selected and is further bisected and so on. This process continues $k-1$ times, leading to k clusters. Each of these bisections is performed so that the resulting two-way clustering solution optimizes a particular criterion function.

Centroid-based approaches, try to assign points to clusters such that the mean square distance of points to the centroid of the assigned cluster is minimized. Centroid-based techniques are suitable only for data in metric space such as Euclidean space in which it is possible to compute a centroid of a given set of points. Medoid-based methods, work with similarity data, i.e., data in an arbitrary similarity space (Ganti et. al, 1999). These techniques try to find medoid as representative points so as to minimize the sum of the distances of points from their closes medoid.

A major drawback of both of these schemes is that they fail for data in which points in given cluster are closer to the center of another cluster than to the center of their own cluster. This can happen in many natural clusters (Han et. al, 1999); for example, if there is a large variation in cluster sizes (as in Figure 3 (a)) or when cluster shapes are convex (an in Figure 3 (b)).



a) Clusters of widely different sizes



b) Clusters with convex shapes

Figure 3 : Data sets on which centroid and medoid approaches fail.

Probabilistic models assume that the data comes from a mixture of several populations whose distributions and priors we want to find. Corresponding algorithms are described in the sub-section *Probabilistic Clustering*. One clear advantage of probabilistic methods is the interpretability of the constructed clusters. Another approach starts with the definition of *objective function* depending on a partition. Iterative optimization partitioning algorithms are subdivided into k-medoids and k-means methods. K-medoid is the most appropriate data point within a cluster that represents it. In k-means case a cluster is represented by its centroid, which is a mean (usually weighted average) of points within a cluster. This works conveniently only with numerical attributes and can be negatively affected

by a single outlier. The corresponding algorithms are reviewed in the sub-sections *K-Medoids Methods* and *K-Means Methods.*

3.1 Probabilistic Clustering

In the *probabilistic* approach, data is considered to be a sample independently drawn from a *mixture model* of several probability distributions (McLachlan and Basford, 1988). The main assumption is that data points are generated by, first, randomly picking a model and, second, by drawing a point x from a corresponding distribution. The area around the mean of each distribution constitutes a natural cluster. The cluster associate with the corresponding distribution's parameters such as mean, variance, etc. Each data point carries not only its observable attributes, but also a hidden cluster ID *(class* in pattern recognition).

Log-likelihood serves as an *objective functio*n, which gives rise to the *Expectation-Maximization* (EM) method. EM is a two-step iterative optimization. Step (E) estimates probabilities , which is equivalent to a soft (fuzzy) reassignment. Step (M) finds an approximation to a mixture model, given current soft assignments. This boils down to finding mixture model parameters that maximize log-likelihood. The process continues until log-likelihood convergence is achieved. Restarting and other tricks are used to facilitate finding better local optimum.

Moore (1999) suggested acceleration of EM method based on a special data index, KD-tree. Data is divided at each node into two descendents by splitting the widest attribute at the center of its range. Each node stores sufficient statistics (including covariance matrix) similar to BIRCH(Zhang et.al, 1996). Approximate computing over a pruned tree accelerates EM iterations. Probabilistic clustering has some important features: (i) it can be modified to handle recodes of complex structure, and (ii) it can be stopped and resumed with consecutive batches of data, since cluster shave representation totally different from sets of points. At any stage of iterative process the intermediate mixture model can be used to assign cases (on-line property) It results in easily interpretable cluster system Because the mixture model has clear probabilistic foundation, the determination of the most suitable number of clusters k becomes a more tractable task. From a data mining perspective, excessive parameter set causes over fitting, while from a probabilistic perspective, number of parameters can be addressed within the Bayesian framework.

Fraley and Raftery, (1999) introduced the algorithm MCLUST. It is a software package (commercially linked with S-PLUS) for hierarchical, mixture model clustering, and discriminant analysis using BIC This algorithm is for estimation of goodness of fit. MCLUST uses Gaussian models with ellipsoids of different volumes, shapes, and orientations. An important property of probabilistic clustering is that mixture model can be naturally generalized to clustering *heterogeneous* data. This is important in practice, where an individual (data object) has multivariate static data (demographics) in combination with variable length dynamic data (customer profile) (Smyth 1999). The dynamic data can consist of finite sequences subject to a first-order Markov model with a transition matrix dependent on a cluster. This framework also covers data objects consisting of *several* sequences, where number n of sequences per is subject to geometric distribution (Cadez et al. 2000). To emulate sessions of different lengths, finite-state Markov model (transitional probabilities between Web site pages) has to be augmented with a special "end" state.

3.2 *K*-Medoids Methods

In *k*-medoids methods a cluster is represented by one of its points. This is an easy solution since it covers any attribute types and that medoids have embedded resistance against outliers since peripheral cluster points do not affect them. When medoids are selected, clusters are defined as subsets of points close to respective medoids, and the objective function is defined as the averaged distance or another dissimilarity measure between a point and its medoid. Representation by k-medoids has two advantages: (i) It presents no limitations on attributes types, and, (ii) the choice of medoids is dictated by the location of a predominant fraction of points inside a cluster and, therefore, it is lesser sensitive to the presence of outliers.

Two early versions of *k*-medoid methods are the algorithm PAM and the algorithm CLARA. PAM is iterative optimization that combines relocation of points between perspective clusters with re-nominating the points as potential medoids. The guiding principle for the process is the effect on an objective function, which, obviously, is a costly strategy. CLARA uses several samples, each with $40+2k$ points, which are each subjected to PAM. The whole dataset is assigned to resulting medoids. Then the objective function is computed, and the best system of medoids is retained.

Further, Ng T. and Han J. (1994) who introduced the algorithm CLARANS in the context of clustering in *spatial* databases. This algorithm is based on random sized search. CLARANS considered a graph whose nodes are the sets of k medoids and an edge connects two nodes if they differ by exactly one medoid. While CLARA compares very few neighbors corresponding to a fixed small sample, CLARANS uses random search to generate neighbors by starting with an arbitrary node and randomly checking *maxneighbour* neighbors. If a neighbor represents a better partition, the process continues with this new node. Otherwise a local minimum is found, and the algorithm restarts until local minima are found. The best node (set of medoids) is returned for the formation of a resulting partition. The weakness of CLARANS is it may not found a real local minimum due to the searching training controlled by maxneighbour.

Ester et al. (1995) extended CLARANS to spatial VLDB. They used R*-trees to relax the original requirement that all the data resides in core memory. This technique allowed *focusing* exploration on the relevant part of the database that resides at a branch of the whole data tree. VLDB improved CLARANS ability by (i) clustering a sample of the dataset that is drawn from each R-Tree data and, (ii) focusing on relevant data points for distance and quality updates.

Zhang T. et.al., (1996) proposed BIRCH that improve and more efficient from previous approach. In BIRCH, each clustering decision is made without scanning all data points or all currently existing clusters. BIRCH is an incremental method that does not require the whole dataset in advance, and only scans the dataset once. It uses measurements that reflect the natural closeness of points and at the same time, can be incrementally maintained doing the clustering process. BIRCH exploits the observation that the data space is usually not uniformly occupied. It used only the data point that is equally important for clustering purpose. In experimentally, BIRCH is shown to perform very well on several large datasets and it significantly superior to CLARANS in terms of quality, speed and order sensitivity. However BIRCH have least reasonable ways of increasing the threshold dynamically. It also cannot adjust dynamically outlier criteria.

3.3 *K*-Means Methods

The ***k-means*** algorithm are proposed by Hartigan and Wong (1979). The name comes from representing each of k clusters C by the mean (or weighted average) c of its points, called centroid. While this obviously does not work well with categorical attributes, it has the good geometric and statistical sense for numerical attributes. The sum of discrepancies between a point and its centroid is used as the objective function. For example, the L_2-norm based objective function, the sum of the squares of errors between the points and the corresponding centroids, is equal to the total intra-cluster variance

$$E(C) = \sum_{j=1:k} \sum_{x_i \in C_j} \| x_i - c_j \|^2 . \quad (3)$$

Consider dataset X cocsisting of data points $x_i = (x_{i1}, \ldots, x_{id}) \in A$ in attribute space A, where $i = 1{:}N$ and each component $x_{il} \in A_l$ is a numerical or nominal categorical attribute. The sum of the squares of errors can be rationalized as (a negative of) log-likelihood for normally distributed mixture model and is widely used in statistics (SSE). Therefore, k-means algorithm can be derived from general probabilistic framework. Note that only means are estimated. A simple modification would normalize individual errors by cluster standard deviation, which makes a lot of sense when clusters have different dispersions. An objective function based on L_2-norm has many unique algebraic properties. Therefore, the cluster separation is achieved simultaneously with the cluster tightness.

There are two versions of k-means iterative optimization. The first version is similar to EM algorithm and consists of two-step major iterations that (i) reassign all the points to their nearest centroids, and (ii) re-compute centroids of newly assembled groups. Iterations continue until a stopping criterion is achieved. This version is known as Forgy's algorithm (Forgy, 1965) and has many advantages:

i. It easily works with any L_p-norm
ii. It allows straightforward parallelization, and
iii. It is insensitive with respect to data ordering.

The second (classic in iterative optimization) version of k-means iterative optimization reassigns points. It based on more detailed analysis of effects on the objective function caused by moving a point from its current cluster to a potentially new one. If a move has a positive effect, the point is relocated and the two centroids are recomputed. It is not clear that this version is computationally feasible, because the outlined analysis requires an inner loop over all member points of involved clusters affected by centroids shifts. However, in L_2 case it is known that all computations can be algebraically reduced to simply computing a single distance. Therefore, in this case both versions have the same computational complexity.

Dhillon et al. (2002) noticed that a Forgy's *spherical* k-means (using cosine similarity instead of Euclidean distance) has a tendency to get stuck when applied to document collections. They noticed that a version reassigning points and immediately re-computing centroids works much better. The k-means algorithm is simple, straightforward, and is based on the firm foundation of analysis of variances. No initialization actually guarantees global minimum for k-means. As is common to any combinatorial optimization, a logical attempt to cure this problem is to use simulated annealing. Zhang (2001) suggested another way to rectify optimization process by soft assignment of points to different clusters with appropriate weights (as EM does), rather than by moving them decisively from one cluster to another. The weights take into

account how well a point fits into recipient cluster. This process involves so-called *harmonic means*.

A generic method to achieve scalability is to preprocess or *squash* the data. Such preprocessing usually also takes care of outliers. It results in approximations that sometimes negatively affect final cluster quality. Pelleg and Moore (1999) suggested how to directly (without any squashing) accelerate *k*-means iterative process by utilizing KD-trees. The algorithm *X*-means also by Pelleg and Moore (2000) goes a step further: in addition to accelerating the iterative process it tries to incorporate a search for the best *k* in the process itself. *X*-means tries to split a part of already constructed cluster based on outcome of BIC criterion. This gives a much better initial guess for the next iteration and covers a user specified range of admissible *k*. The tremendous popularity of *k*-means algorithm has brought to life many other extensions and modifications.

4 CLUSTER SELECTION SCHEME

The tree basic criteria to determine which pair of clusters to be merged next are single-link, complete-link and group average (Zhao and Karypis, 2001). The single link method represented each cluster by all the data points in cluster (Jain and Dubes, 1988). The similarity between two clusters is measured by the similarity of the closest pair of data points belonging to different clusters. Unlike the centroid/medoid based methods, this method can find clusters of arbitrary shape and different sizes. The single link method measures the similarity of two clusters by the maximum similarity between any pair of objects from each cluster, whereas the complete-link uses the minimum similarity. That is, the similarity between two clusters S_i and S_j is given by

$$\text{sim}_{\text{single-link}}(S_i, S_j) = \max_{d_i \in S_i,\, d_j \in S_j} \{\cos(d_i, d_j)\}. \tag{4}$$

The complete link scheme uses minimum similarity to measure the same similarity. That is,

$$\text{sim}_{\text{single-link}}(S_i, S_j) = \min_{d_i \in S_i,\, d_j \in S_j} \{\cos(d_i, d_j)\}. \tag{5}$$

The group average scheme (UPGMA) overcomes these problems by measuring the similarity of two clusters as the average of the pairwise similarity from each cluster. That is,

$$\text{sim}_{\text{UPGMA}}(S_i, S_j) = \frac{1}{n_i n_j} \sum_{d_i \in S_i,\, d_j \in S_j} \cos(d_i, d_j) = \frac{D_i^t D_j}{n_i n_j} \tag{6}$$

In general, both the single- and complete-link approaches do not work very well because they either base their decisions to a limited amount of information (single link), or assume that all the objects in the cluster are very similar to each other (complete-link). On the other hand , the group average approach measures the similarity of two clusters by the average of the pair-wise similarity of the objects from each cluster and does not suffer from the problems arising with single- and complete-link.

5 CLUSTERING CRITERION FUNCTIONS

There are many criterion functions have been proposed. Criterion functions used in the partitional clustering reflect the underlying definition of the 'goodness' of clusters. Clustering criterion function can be classified into four groups: internal, external, hybrid and graph-based. A key characteristic of most partitional clustering algorithms is that they use a global criterion function whose optimization drives the entire clustering process (Zhao and Karypis, 2002). Criterion functions measure various aspects of intra-cluster similarity, inter-cluster dissimilarity and their combinations. These criterion functions utilize different views of the underlying collection, by either modeling the objects as vectors in high-dimensional space, or by modeling the collection as a graph. The partitional clustering can be considered as an optimization procedure that tries to create high quality clusters according to a particular criterion function. (Zhao et. al, 2001)

5.1 Internal Criterion Functions

The internal criterion functions focus on producing a clustering solution that optimizes a function defined only over the objects of each cluster and does not take into account the objects assigned to different clusters (Zhao and Karypis, 2002). The first internal criterion function maximizes the sum of the average pairwise similarities between the objects assigned to each cluster, weighted according to the size of each cluster. Specially, if cosine function is used to measure the similarity between objects, then the clustering solution must optimize the following criterion function:

$$\text{maximize } \chi_1 = \sum_{r=1}^{k} n_r \left(\frac{1}{n_r^2} \sum_{d_i,d_j \in S_r} \cos(d_i,d_j) \right) = \sum_{r=1}^{k} \frac{\|D_r\|^2}{n_r}, \qquad (7)$$

In the second criterion function, each cluster is represent by its centroid vector and the goal is to find the clustering solution that maximizes the similarity between each object and a centroid. This algorithm is used by the popular vector-space variant of the *K*-means algorithm. The criterion function becomes the following :

$$\text{maximize } \chi_2 = \sum_{r=1}^{k} \sum_{d_j \in S_r} xxx(d_i, C_r) = \sum_{r=1}^{k} \sum_{d_j \in S_r} \frac{d_i^t C_r}{\|C_r\|} = \sum_{r=1}^{k} \frac{D_r^t C_r}{\|C_r\|} = \sum_{r=1}^{k} \frac{D_r^t D_r}{\|D_r\|} = \sum_{r=1}^{k} \| \|. \qquad (8)$$

5.2 External Criterion Function

It is quite hard to define external criterion functions that lead to meaningful solution. The external criterion function derive the clustering solution by focusing on optimizing a function that is based on how the various clusters are different from each other. For many problems, criterion function has trivial solutions that can be achieved by assigning to the first $k-1$ clusters a single object that shares very few terms with the rest, and then assigning the rest of the objects to the kth cluster. This criterion function tries to separate the objects of each cluster from the entire collection, as opposed trying to separate the objects among different clusters. This external criterion function was motivated by multiple discriminant analysis and is similar to minimizing the trace of the between-cluster scatter matrix.(Duda, 2001). The external criterion function is defined as

$$\text{minimize } \sum_{r=1}^{k} n_r \cos(C_r, C), \qquad (9)$$

Where C is the centroid vector of the entire collection. By minimizing the cosine between the centroid vector of each cluster to the centroid vector of the entire collection we essentially try to increase the angle between them as much as possible. The contribution of each cluster is weighted based on the cluster size, so that larger clusters will weight heavier in the overall clustering solution. The next equation can be written as

$$\sum_{r=1}^{k} n_r \cos(C_r, C) = \sum_{r=1}^{k} n_r \frac{C_r^t C}{\|C_r\| \|C\|} = \sum_{r=1}^{k} n_r \frac{D_r^t D}{\|D_r\| \|D\|} = \frac{1}{\|D\|}\left(\sum_{r=1}^{k} n_r \frac{D_r^t D}{\|D_r\|}\right) \tag{10}$$

Where D is the composite vector of the entire object collection. Since 1/ ||D|| is constant the clustering solution the criterion function can be re-stated as :

$$\text{minimize } \mathcal{E}_1 = \sum_{r=1}^{k} n_r \frac{D_r^t D}{\|D_r\|} \tag{11}$$

5.3 Hybrid Criterion Function

The hybrid criterion functions simultaneously optimize multiple individual criterion function. $\mathcal{E}_1$ is actually a hybrid criterion function that combines both external and internal characteristics of the cluster. This type of hybrid criterion function are obtain by combining criterion χ_1 with $\mathcal{E}_1$ and χ_2 with $\mathcal{E}_1$.,

$$\text{maximize } \mathcal{H}_1 = \frac{\chi_1}{\mathcal{E}_1} = \frac{\sum_{r=1}^{k} \|D_r\|^2 / n_r}{\sum_{r=1}^{k} n_r D_r^t D / \|D_r\|} \tag{12}$$

And the second is

$$\text{maximize } \mathcal{H}_2 = \frac{\chi_1}{\mathcal{E}_1} = \frac{\sum_{r=1}^{k} \|D_r\|^2}{\sum_{r=1}^{k} n_r D_r^t D / \|D_r\|} \tag{13}$$

6 CONCLUSION

Clustering is one of the popular and important techniques in data mining. An ideal clustering algorithm should be the one that can automatically discover the natural clusters present in the dataset based on the underlying cluster definition. However, there are no such universal cluster definitions and clustering algorithms suitable for all kinds of datasets. Different clustering algorithms with various parameters or initial conditions with same dataset will give different clusters. It is essential to know whether the resulting clusters are valid and how to compare the quality of the clustering results, so that the right clustering algorithm can be chosen and the best clustering result can be used for further analysis. As result, most existing algorithms require either the number of clusters to be provided as parameter as it is done in the case of K-means, or a similarity threshold that will be used to terminate the merging process in the case of agglomerative clustering.

Existing clustering algorithms for sequence and structure datasets operate on the object's similarity space. Algorithm such as feature- and similarity- based clustering algorithm are quite limiting as they cannot scale to very large datasets, cannot be used to provide a description as to

why a set of objects was assigned to the same cluster that is native to the object's features and cannot be used to find clusters that have conserved features. Furthermore it also cannot provide a description as to why the objects assigned to the same cluster that is native to the object's feature.

The only way to overcome this shortcoming is to : (i) Develop scalable and computationally algorithms for large sequence and structure datasets that operates directly on the object's native representation. (ii) Develop clustering algorithms that can provide concise explanation on the characteristics of the objects that were assigned to each cluster.

REFERENCES

1. Aggarwal, C.C., Hinneburg, A., and Keim, D.A., *On the surprising behavior of distance metrics in high dimensional space. IBM Research report*, RC 21739, 2000..
2. Berkhin, P. and Becher, J. Learning Simple Relations: Theory and Applications.In *Proceedings of the 2nd SIAM ICDM*, 420-436, Arlington, VA, 2002.
3.
4. Defays, D. An efficient algorithm for a complete link method. *The Computer Journal*, 20, 364-366, 1977.
5. Dhillon, I., Mallela, S., and Kumar, R. Enhanced Word Clustering for Hierarchical Text Classification, In *Proceedings of the 8th ACM SIGKDD*, 191-200, Edmonton, Canada, 2002.
6. Duda R.O., Hact PE. and Stork D.G. *Pattern Classification*. John Willey & Sons, 2001
7. Ester, M., Kriegel, H-P., and Xu, X. A database interface for clustering in large spatial databases. In *Proceedings of the 1st ACM SIGKDD*, 94-99, Montreal, Canada, 1995.
8. Forgy, E. Cluster analysis of multivariate data: Efficiency versus interpretability of classification. *Biometrics*, 21, 768-780, 1965.
9. Fraley, C. And RAFTERY, A. MCLUST: Software for model-based cluster and discriminant analysis, *Tech Report 342*, Dept. Statistics, Univ. of Washington, 1999.
10. Ganti V., Ramakrishnan R., Gerkhe J., Powell A. and French J. Clustering large datasets in arbitary metric spaces. In *Proc. Of the 15th International Conf. On Data Eng.*, 1999.
11. Han, E-H., Karypis, G., Kumar, V., and Mobasher, B. Clustering based on association rule hypergraphs. *ACM SIGMOD Conference, Data Mining Workshop*(DMKD'97), Tucson, Arizona, 1997.
12. Hartigan, J. And Wong, M. Algorithm AS136: A k-means clustering algorithm. *Applied Statistics*, 28, 100-108, 1979.
13. Jain, A. And Dubes, R. *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, NJ, 1988.
14. Karypis, G. and Kumar, V., A fast and highly quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20, 1.
15. Karypis, G.;Han, E.-H.; and Kumar. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer: Special Issue on Data Analysis and Mining* 32(8):68-75,1999.
16. Kaufman, L. And Rousseeuw, P. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, New York, NY, 1990.
17. Lance, G. and Williams W. 1967. A general theory of classification sorting strategies. *Computer Journal*, 9, 373-386, 1999.
18. Mclachlan, G. and Basford, K. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York, NY, 1988.

19. Ng, R. and Han, J. Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th Conference on VLDB*, 144-155, Santiago, Chile, 1994.
20. Olson, C. Parallel algorithms for hierarchical clustering. *Parallel Computing*, 21, 1313-1325, 1995.
21. Pelleg, D. and Moore, A. Accelerating exact k-means algorithms with geometric reasoning. In *Proceedings of the 5th ACM SIGKDD*, 277-281, San Diego, CA, 1999.
22. Pelleg, D. and Moore, A. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *Proceedings 17th ICML*, Stanford University, 2000.
23. Sibson, R. Slink: An optimally efficient algorithm for the single link cluster method. *Computer Journal*, 16, 30-34, 1973.
24. Smyth, P. Probabilistic model-based clustering of multivariate and sequential data. In *Proceedings of the 7th International Workshop on AI and Statistics*, 299-304, 1999.
25. Stonebraker M., Agrawal R., Dayal U., Neuhold E.J. and Reuter A. DBMS research at a crossroads : The vienna update. In *Proc. Of the 19th VLDB Conference*, pages 688-698, Dublin, Ireland, 1993.
26. Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: An efficient clustering algorithm for large databases. In *Proc. Of 1998 ACM-SIGMOD Int.Conf. on Management of Data*, 1998.
27. Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. ROCK: a robust clustering algorithm for categorical attributes. In *Proc. Of the 15th Int'l Conf. On Data Eng.*, 1999.
28. Tung, A.K.H., Hou, J., And Han, J. Spatial clustering in the presence of obstacles. In *Proceedings of the 17th ICDE*, 359-367, Heidelberg, Germany. 2001.
29. Voorhees, E.M. Implementing agglomerative hierarchical clustering algorithms for use in document retrieval. *Information Processing and Management*, 22, 6, 465-476, 1986.
30. Y. Zhao ang G. Karypis. Comparison of agglomerative and partitional document clustering algorithms. In *SIAM(2002) workshop on Clustering High-dimensional Data and Its Applications*, 2002.
31. Y. Zhao ang G. Karypis. Criterion functions for document clustering: Experiments and analysis. Technical report TR #01-40, Department of Computer Science, University of Minnesota, Minneapolis, MN, 2001.
32. Zhang, B. Generalized k-harmonic means – dynamic weighting of data in unsupervised learning. In *Proceedings of the 1st SIAM ICDM*, Chicago, IL. 2001.
33. Zhang, T., Ramakrishnan, R., and LIVNY, M. BIRCH: A new data clustering algorithm and its applications. *Journal of Data Mining and Knowledge Discovery*, 1, 2, 141-182. 1997.