An overview of clustering methods

Article *in* Intelligent Data Analysis · January 2007
Source: DBLP

3 authors:

Mahamed Omran
Gulf University for Science and Technology (Kuwait)
61 PUBLICATIONS **2,592** CITATIONS
SEE PROFILE

Andries Engelbrecht
University of Pretoria
300 PUBLICATIONS **12,974** CITATIONS
SEE PROFILE

Ayed A. Salman
Kuwait University
52 PUBLICATIONS **2,253** CITATIONS
SEE PROFILE

An Overview of Clustering Methods

Short title: Clustering Methods

Mahamed G.H. Omran,[1] Andries P Engelbrecht[1,3] and Ayed Salman[2]

[1]Department of Computer Science, School of Information Technology

University of Pretoria, Pretoria 0002, South Africa

[2]Department of Computer Engineering, Kuwait University, Kuwait

[3]Department of Computer Science, School of Information Technology

University of Pretoria, Pretoria 0002, South Africa

Tel: +27 12 420 3578 Fax: +27 12 362 5188

E-mail: engel@cs.up.ac.za

Abstract— Data clustering is the process of identifying natural groupings or clusters within multidimensional data based on some similarity measure. Clustering is a fundamental process in many different disciplines. Hence, researchers from different fields are actively working on the clustering problem. This paper provides an overview of the different representative clustering methods. In addition, several clustering validations indices are shown. Furthermore, approaches to automatically determine the number of clusters are presented. Finally, application of different heuristic approaches to the clustering problem is also investigated.

Keywords: Clustering; Clustering Validation; Hard Clustering; Fuzzy Clustering; Unsupervised Learning

1. Introduction

Data clustering is the process of identifying natural groupings or clusters within multidimensional data based on some similarity measure (e.g. Euclidean distance) [Jain *et al*. 1999; Jain *et al*. 2000]. It is an important process in pattern recognition and machine learning [Hamerly and Elkan 2002]. Furthermore, data clustering is a central process in Artificial Intelligence (AI) [Hamerly 2003]. Clustering algorithms are used in many applications, such as image segmentation [Coleman and Andrews 1979; Jain and Dubes 1988; Turi 2001], vector and color image quantization [Kaukoranta *et al*. 1998; Baek *et al*. 1998; Xiang 1997], data mining [Judd *et al*. 1998], compression [Abbas and Fahmy 1994], machine learning [Carpineto and Romano 1996], etc. A cluster is usually identified by a cluster center (or *centroid*) [Lee and Antonsson 2000]. Data clustering is a difficult problem in unsupervised pattern recognition as the clusters in data may have different shapes and sizes [Jain *et al*. 2000].

Due to the prohibitive amount of research conducted in the area of clustering, a survey paper investigating the *state-of-the-art* clustering methods is generally welcomed. Hence, the purpose of this paper is to provide such an overview of representative clustering methods. However, trying to address all the clustering

methods on one paper is not possible. Therefore, this paper tries to provide the reader with an overview of a set of representative clustering methods.

The reminder of this paper is organized as follows: Section 2 provides a background material. Section 3 surveys different clustering techniques. Several clustering validation techniques are presented in Section 4. Methods for determining the number of clusters in a data set are given in Section 5. Section 6 provides a brief introduction to the use of Self-Organizing Maps for clustering. Clustering using stochastic techniques is investigated in Section 7. Finally, Section 8 concludes the paper.

2. Backgrounds

This section defines the terms used throughout the paper and it provides the reader with the necessary background material to follow-up the discussion in the paper.

2.1. Definitions

The following terms are used in this paper:

- A *pattern* (or *feature vector*), $\boldsymbol{z}$, is a single object or data point used by the clustering algorithm [Jain *et al*. 1999].
- A *feature* (or *attribute*) is an individual component of a pattern [Jain *et al*. 1999].
- A *cluster* is a set of similar patterns, and patterns from different clusters are not similar [Everitt 1974].
- *Hard* (or *Crisp*) clustering algorithms assign each pattern to one and only one cluster.
- *Fuzzy* clustering algorithms assign each pattern to each cluster with some degree of membership.
- A *distance measure* is a metric used to evaluate the similarity of patterns [Jain *et al*. 1999].

The clustering problem can be formally defined as follows (Veenman *et al*. 2003):

Given a data set $\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_p, \ldots, \boldsymbol{z}_{N_p}\}$ where $\boldsymbol{z}_p$ is a pattern in the N_d-dimensional feature space, and N_p is the number of patterns in $\boldsymbol{Z}$, then the clustering of $\boldsymbol{Z}$ is the partitioning of $\boldsymbol{Z}$ into K clusters $\{\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_K\}$ satisfying the following conditions:

- Each pattern should be assigned to a cluster, i.e.
  $\cup_{k=1}^{K} \boldsymbol{C}_k = \boldsymbol{Z}$
- Each cluster has at least one pattern assigned to it, i.e.
  $\boldsymbol{C}_k \neq \phi, \quad k = 1, \ldots, K$
- Each pattern is assigned to one and only one cluster (in case of hard clustering only), i.e.
  $\boldsymbol{C}_k \cap \boldsymbol{C}_{kk} = \phi \quad \text{where } k \neq kk$

2.2. Similarity Measures

As previously mentioned, clustering is the process of identifying natural groupings or clusters within multidimensional data based on some similarity measure. Hence, similarity measures are fundamental components in most clustering algorithms [Jain *et al*. 1999].

The most popular way to evaluate a similarity measure is the use of distance measures. The most widely used distance measure is the Euclidean distance defined as

$$d(\boldsymbol{z}_u, \boldsymbol{z}_w) = \sqrt{\sum_{j=1}^{N_d} (z_{u,j} - z_{w,j})^2} = \|\boldsymbol{z}_u - \boldsymbol{z}_w\| \quad (1)$$

Euclidean distance is a special case (when $\alpha = 2$) of the Minkowski metric [Jain *et al*. 1999] defined as

$$d^{\alpha}(\boldsymbol{z}_u, \boldsymbol{z}_w) = (\sum_{j=1}^{N_d} (z_{u,j} - z_{w,j})^{\alpha})^{1/\alpha} = \|\boldsymbol{z}_u - \boldsymbol{z}_w\|^{\alpha} \quad (2)$$

When $\alpha = 1$, the measure is referred to as the Manhattan distance [Hamerly 2003].

Clustering data of high dimensionality using the Minkowski metric is usually not efficient because the distance between the patterns increases with increase in dimensionality. Hence, the concepts of near and far become weaker [Hamerly 2003]. Furthermore, for the Minkowski metric, the largest-scaled feature tends to dominate the other features. This can be solved by normalizing the features to a common range [Jain *et al*. 1999]. One way to do this is by using the cosine distance (or vector dot product) which is the sum of the product of each component from two vectors defined as

$$< \boldsymbol{z}_u, \boldsymbol{z}_w > = \frac{\sum_{j=1}^{N_d} z_{u,j} z_{w,j}}{\|\boldsymbol{z}_u\| \|\boldsymbol{z}_w\|} \quad (3)$$

where $< \boldsymbol{z}_u, \boldsymbol{z}_w > \in [-1,1]$.

The cosine distance is actually not a distance but rather a similarity metric. In other words, the cosine distance measures the difference in the angle between two vectors not the difference in the magnitude between two vectors. The cosine distance is suitable for clustering data of high dimensionality [Hamerly 2003].

Another distance measure is the Mahalanobis distance defined as

$$d_{\mathrm{M}}(\boldsymbol{z}_u, \boldsymbol{z}_w) = (\boldsymbol{z}_u - \boldsymbol{z}_w)\Sigma^{-1}(\boldsymbol{z}_u - \boldsymbol{z}_w)^{\mathrm{T}} \quad (4)$$

where Σ is the covariance matrix of the patterns. The Mahalanobis distance gives different features different weights based on their variances and pairwise linear correlations. Thus, this metric implicitly assumes that the densities of the classes are multivariate Gaussian [Jain *et al*. 1999].

3. Clustering Techniques

Most clustering algorithms are based on two popular techniques known as *hierarchical* and *partitional* clustering [Frigui and Krishnapuram 1999; Leung *et al*. 2000]. In the following, an overview of both techniques is presented with an elaborate discussion of popular hierarchical and partitional clustering algorithms.

3.1. Hierarchical Clustering Techniques

Algorithms in this category generate a cluster tree (or *dendrogram*) by using heuristic splitting or merging techniques [Hamerly 2003]. A cluster tree is defined as "a tree showing a sequence of clustering with each clustering being a partition of the data set" [Leung *et al*. 2000]. Algorithms that use splitting to generate the cluster tree are called *divisive*. On the other hand, the more popular algorithms that use merging to generate the cluster tree are called *agglomerative*. Divisive hierarchical algorithms start with all the patterns assigned to a single cluster. Then, splitting is applied to a cluster in each stage until each cluster consists of one pattern. Contrary to divisive hierarchical algorithms, agglomerative hierarchical algorithms start with each pattern assigned to one cluster. Then, the two most similar clusters are merged together. This step is repeated until all the patterns are assigned to a single cluster [Turi 2001]. Several agglomerative hierarchical algorithms were proposed in the literature which differ in the way that the two most similar clusters are calculated. The two most popular agglomerative hierarchical algorithms are the *single link* [Sneath and Sokal 1973] and *complete link* [Anderberg 1973] algorithms. Single link algorithms merge the clusters whose distance between their closest patterns is the smallest. Complete link algorithms, on the other hand, merge the clusters whose distance between their most distant patterns is the smallest [Turi 2001]. In general, complete link algorithms generate compact clusters while single link algorithms generate elongated clusters. Thus, complete link algorithms are generally more useful than single link algorithms [Jain *et al*. 1999]. Another less popular agglomerative hierarchical algorithm is the *centroid* method [Anderberg 1973]. The centroid algorithm merges the clusters whose distance between their centroids is the smallest. One disadvantage of the centroid algorithm is that the characteristic of a very small cluster is lost when merged with a very large cluster [Turi 2001]. More details about traditional hierarchical clustering techniques can be found in Everitt [1974].

Recently, a hierarchical clustering approach to simulate the human visual system by modeling the blurring effect of lateral retinal interconnections based on scale space theory has been proposed by Leung *et al*. [2000]. The following paragraph provides the reader with a good idea about this approach as described by Leung *et al*. [2000]:

> "In this approach, a data set is considered as an image with each light point located at a datum position. As we blur this image, smaller light blobs merge into larger ones until the whole image becomes one light blob at a low level of resolution. By identifying each blob with a cluster, the blurring process generates a family of clustering along the hierarchy."

According to Leung *et al*. [2000], this approach has several advantages, including:

- it is not sensitive to initialization,

- it is robust in the presence of noise in the data set, and
- it generates clustering that is similar to that perceived by human eyes.

In general, hierarchical clustering techniques have the following advantages [Frigui and Krishnapuram 1999]:

- the number of clusters need not to be specified *a priori*, and
- they are independent of the initial conditions.

However, hierarchical clustering techniques generally suffer from the following drawbacks:

- They are computationally expensive (time complexity is $O(N_p^2 \log N_p)$ and space complexity is $O(N_p^2)$ [Turi 2001]). Hence, they are not suitable for very large data sets.
- They are static, i.e. patterns assigned to a cluster cannot move to another cluster.
- They may fail to separate overlapping clusters due to a lack of information about the global shape or size of the clusters.

3.2. Partitional Clustering Techniques

Partitional clustering algorithms divide the data set into a specified number of clusters. These algorithms try to minimize certain criteria (e.g. a square error function) and can therefore be treated as optimization problems. However, these optimization problems are generally NP-hard and combinatorial [Leung *et al*. 2000]. The advantages of hierarchical algorithms are the disadvantages of the partitional algorithms and *vice versa*. Because of their advantages, partitional clustering techniques are more popular than hierarchical techniques in pattern recognition [Jain *et al*. 2000], hence, this paper concentrates on partitional techniques.

Partitional clustering algorithms are generally iterative algorithms that converge to local optima [Hamerly and Elkan 2002]. Employing the general form of iterative clustering used by Hamerly and Elkan [2002], the steps of an iterative clustering algorithm are:

1. Randomly initialize the K cluster centroids
2. **Repeat**
   - **For** each pattern, $\boldsymbol{z}_p$, in the data set **do**
     Compute its membership $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p)$ to each centroid $\boldsymbol{m}_k$ and its weight $w(\boldsymbol{z}_p)$
   - **endloop**
   - Recalculate the K cluster centroids, using

$$\boldsymbol{m}_k = \frac{\sum_{\forall \boldsymbol{z}_p} u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) w(\boldsymbol{z}_p) \boldsymbol{z}_p}{\sum_{\forall \boldsymbol{z}_p} u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) w(\boldsymbol{z}_p)} \tag{5}$$

until a stopping criterion is satisfied.

In the above algorithm, $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p)$ is the membership function which quantifies the membership of pattern $\boldsymbol{z}_p$ to cluster k. The membership function, $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p)$, must satisfy the following constraints:

- $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) \geq 0, \; p = 1,\ldots, N_p$ and $k = 1,\ldots, K$
- $\sum_{k=1}^{K} u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = 1, \; p = 1,\ldots, N_p$

Crisp clustering algorithms use a *hard* membership function (i.e. $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) \in \{0,1\}$), while fuzzy clustering algorithms use a *soft* member function (i.e. $u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) \in [0,1]$) [Hamerly and Elkan 2002].

The weight function, $w(\boldsymbol{z}_p)$, in Eq. (5) defines how much influence pattern $\boldsymbol{z}_p$ has in recomputing the centroids in the next iteration, where $w(\boldsymbol{z}_p) > 0$ [Hamerly and Elkan 2002]. The weight function was proposed by Zhang [2000].

Different stopping criteria can be used in an iterative clustering algorithm, for example:

- stop when the change in centroid values are smaller than a user-specified value,
- stop when the quantization error is small enough, or
- stop when a maximum number of iterations has been exceeded.

In the following, popular iterative clustering algorithms are described by defining the membership and weight functions in Eq. (5).

3.2.1. The K-means Algorithm

The most widely used partitional algorithm is the iterative K-means approach [Forgy 1965]. The objective function that the K-means optimizes is

$$J_{\text{K-means}} = \sum_{k=1}^{K} \sum_{\forall \boldsymbol{z}_p \in \boldsymbol{C}_k} d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) \tag{6}$$

Hence, the K-means algorithm minimizes the intra-cluster distance [Hamerly and Elkan 2002]. The K-means algorithm starts with K centroids (initial values for the centroids are randomly selected or derived from *a priori* information). Then, each pattern in the data set is assigned to the closest cluster (i.e. closest centroid). Finally, the centroids are recalculated according to the associated patterns. This process is repeated until convergence is achieved.

The membership and weight functions for K-means are defined as

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \begin{cases} 1 & \text{if } d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) = \arg\min_k \left\{ d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) \right\} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$w(\boldsymbol{z}_p) = 1 \tag{8}$$

Hence, K-means has a hard membership function. Furthermore, K-means has a constant weight function, thus, all patterns have equal importance [Hamerly and Elkan 2002].

The K-means algorithm has the following main advantages [Turi 2001]:

- it is very easy to implement, and
- its time complexity is O(N_p) making it suitable for very large data sets.

However, the K-means algorithm has the following drawbacks [Davies 1997]:

- the algorithm is data-dependent,
- it is a greedy algorithm that depends on the initial conditions, which may cause the algorithm to converge to suboptimal solutions, and
- the user needs to specify the number of clusters in advance.

The K-medoids algorithm is similar to K-means with one major difference, namely, the centroids are taken from the data itself [Hamerly 2003]. The objective of K-medoids is to find the most centrally located patterns within the clusters [Halkidi *et al*. 2001]. These patterns are called *medoids*. Finding a single medoid requires $O(N_p^2)$. Hence, K-medoids is not suitable for moderately large data sets.

3.2.2. The Fuzzy C-means Algorithm

A fuzzy version of K-means, called Fuzzy C-means (FCM) (sometimes called fuzzy K-means), was proposed by Bezdek [1980; 1981]. FCM is based on a fuzzy extension of the least-square error criterion. The advantage of FCM over K-means is that FCM assigns each pattern to each cluster with some degree of membership (i.e. fuzzy clustering). This is more suitable for real applications where there are some overlaps between the clusters in the data set. The objective function that the FCM optimizes is

$$J_{\text{FCM}} = \sum_{k=1}^{K}\sum_{p=1}^{N_p} u_{k,p}^{q} d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) \tag{9}$$

where q is the fuzziness exponent, with $q \geq 1$. Increasing the value of q will make the algorithm more fuzzy; $u_{k,p}$ is the membership value for the p^{th} pattern in the k^{th} cluster satisfying the following constraints:

- $u_{k,p} \geq 0\,,\ p = 1,\ldots, N_p$ and $k = 1,\ldots, K$
- $\sum_{k=1}^{K} u_{k,p} = 1\,,\ p = 1,\ldots, N_p$

The membership and weight functions for FCM are defined as [Hamerly and Elkan 2002]

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \frac{\left\|\boldsymbol{z}_p - \boldsymbol{m}_k\right\|^{-2/(q-1)}}{\sum_{k=1}^{K}\left\|\boldsymbol{z}_p - \boldsymbol{m}_k\right\|^{-2/(q-1)}} \tag{10}$$

$$w(\boldsymbol{z}_p) = 1 \tag{11}$$

Hence, FCM has a soft membership function and a constant weight function. In general, FCM performs better than K-means [Hamerly 2003] and it is less affected by the presence of uncertainty in the data [Liew *et al*. 2000]. However, as in K-means it requires the user to specify the number of clusters in the data set. In addition, it may converge to local optima [Jain *et al*. 1999].

Krishnapuram and Keller [1993; 1996] proposed a possibilistic clustering algorithm, called *possibilistic* C-means. Possibilistic clustering is similar to fuzzy clustering; the main difference is that in possibilistic clustering the membership values may not sum to one [Turi 2001]. Possibilistic C-means works well in the presence of noise in the data set. However, it has several drawbacks, namely [Turi 2001],

- it is likely to generate coincident clusters,
- it requires the user to specify the number of clusters in advance,
- it converges to local optima, and
- it depends on initial conditions.

3.2.3. The Gaussian Expectation-Maximization Algorithm

Another popular clustering algorithm is the Expectation-Maximization (EM) algorithm [McLachlan and Krishnan 1997; Rendner and Walker 1984; Bishop 1995]. EM is used for parameter estimation in the presence of some unknown data [Hamerly 2003]. EM partitions the data set into clusters by determining a mixture of Gaussians fitting the data set. Each Gaussian has a mean and covariance matrix [Alldrin *et al*. 2003]. The objective function that the EM optimizes as defined by Hamerly and Elkan [2002] is

$$J_{\text{EM}} = -\sum_{p=1}^{N_p} \log\left(\sum_{k=1}^{K} p(\boldsymbol{z}_p \mid \boldsymbol{m}_k)\, p(\boldsymbol{m}_k)\right) \tag{12}$$

where $p(\boldsymbol{z}_p \mid \boldsymbol{m}_k)$ is the probability of $\boldsymbol{z}_p$ given that it is generated by a Gaussian distribution with centroid $\boldsymbol{m}_k$, and $p(\boldsymbol{m}_k)$ is the prior probability of centroid $\boldsymbol{m}_k$.

The membership and weight functions for EM are defined as [Hamerly and Elkan 2002]

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \frac{p(\boldsymbol{z}_p \mid \boldsymbol{m}_k)\, p(\boldsymbol{m}_k)}{p(\boldsymbol{z}_p)} \tag{13}$$

$$w(\boldsymbol{z}_p) = 1 \tag{14}$$

Hence, EM has a soft membership function and a constant weight function. The algorithm starts with an initial estimate of the parameters. Then, an *expectation* step is applied where the known data values are used to compute the expected values of the unknown data [Hamerly 2003]. This is followed by a *maximization* step where the known and expected values of the data are used to generate a new estimate of the parameters. The expectation and maximization steps are repeated until convergence.

Results from Veenman *et al*. [2002] and Hamerly [2003] showed that K-means performs comparably to EM. Furthermore, Aldrin *et al*. [2003] stated that EM fails on high-dimensional data sets due to numerical precision problems. They also observed that Gaussians often collapsed to delta functions [Alldrin *et al*. 2003]. In addition, EM depends on the initial estimate of the parameters [Hamerly 2003; Turi 2001] and it requires the user to specify the number of clusters in advance. Moreover, EM assumes that the density of each cluster is Gaussian which may not always be true [Ng *et al*. 2001].

3.2.4. The K-harmonic Means Algorithm

Recently, Zhang and colleagues [1999; 2000] proposed a novel algorithm called K-harmonic means (KHM), with promising results. In KHM, the harmonic mean of the distance of each cluster center to every pattern is computed. The cluster centroids are then updated accordingly. The objective function that the KHM optimizes is

$$J_{\text{KHM}} = \sum_{p=1}^{N_p} \frac{K}{\sum_{k=1}^{K} \frac{1}{\left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{\alpha}}} \tag{15}$$

where α is a user-specified parameter, typically $\alpha \geq 2$.

The membership and weight functions for KHM are [Hamerly and Elkan 2002]

$$u(\boldsymbol{m}_k \mid \boldsymbol{z}_p) = \frac{\left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha-2}}{\sum_{k=1}^{K} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha-2}} \tag{16}$$

$$w(\boldsymbol{z}_p) = \frac{\sum_{k=1}^{K} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha-2}}{\left(\sum_{k=1}^{K} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^{-\alpha} \right)^2} \tag{17}$$

Hence, KHM has a soft membership function and a varying weight function. KHM assigns higher weights for patterns that are far from all the centroids to help the centroids in covering the data [Hamerly and Elkan 2002].

Contrary to K-means, KHM is less sensitive to initial conditions and does not have the problem of collapsing Gaussians exhibited by EM [Alldrin *et al*. 2003]. Experiments conducted by Zhang *et al*. [1999], Zhang [2000] and Hamerly and Elkan [2002] showed that KHM outperformed K-means, FCM (according to Hamerly and Elkan [2002]) and EM.

3.2.5. Hybrid 2

Hamerly and Elkan [2002] proposed a variation of KHM, called Hybrid 2 (H2), which uses the soft membership function of KHM (i.e. Eq. (16)) and the constant weight function of K-means (i.e. Eq. (8)). Hamerly and Elkan [2002] showed that H2 outperformed K-means, FCM and EM. However, KHM, in general, performed slightly better than H2.

K-means, FCM, EM, KHM and H2 are linear time algorithms (i.e. their time complexity is $O(N_p)$) making them suitable for very large data sets. According to Hamerly [2003], FCM, KHM and H2 - all use soft membership functions - are the best available clustering algorithms.

3.3. Non-iterative Partitional Algorithms

Another category of unsupervised partitional algorithms includes the non-iterative algorithms. The most widely used non-iterative algorithm is MacQueen's K-means algorithm [MacQueen 1967]. This algorithm works in two phases: the first phase finds the centroids of the clusters, and the second clusters the patterns. Competitive Learning (CL) updates the centroids sequentially by moving the closest centroid toward the pattern being classified [Scheunders 1997]. These algorithms suffer the drawback of being dependent on the order in which the data points are presented. To overcome this problem, data points are presented in a random order [Davies 1997]. In general, iterative algorithms are more effective than non-iterative algorithms, since they are less dependent on the order in which data points are presented.

3.4. Other Clustering Techniques

Another type of clustering algorithms includes the *Nearest Neighbor* clustering algorithm proposed by Lu and Fu [1978]. For each unclassified pattern, the algorithm finds the nearest classified pattern whose distance from the unclassified pattern is less than a pre-specified threshold. The unclassified pattern is then assigned to the cluster of the classified pattern. This process is repeated until all the patterns become classified or no further assignments can occur [Jain *et al*. 1999].

Recently, a new type of clustering algorithms called *spectral* clustering algorithms [Ng *et al*. 2001; Bach and Jordan 2003] has been proposed by computer vision researchers and graph theorists. Spectral clustering is based on spectral graph theory [Chung 1997] where a graph representing the data (the graph is analogous to a matrix of the distance between the patterns in the data set) is searched by the spectral clustering algorithm for globally optimal cuts [Hamerly 2003]. One major advantage of spectral clustering is that it can generate arbitrary-shaped clusters. However, spectral clustering suffers from two major drawbacks [Hamerly 2003]:

- It is computationally expensive (its time complexity is $O(N_p^3 + N_d N_p^2)$). Hence, they are not suitable for moderately large data sets.
- It requires the user to specify a kernel width parameter which has a profound effect on the result of the spectral clustering algorithm. Choosing a good value for this parameter is usually difficult.

The *mean shift* algorithm [Comaniciu and Meer 2002] also automatically finds the number of clusters in a data set and can work with arbitrary shaped clusters. The mean shift algorithm starts with a number of kernel estimators in the input space. These estimators are then repeatedly moved towards areas of higher density. When all the kernels reached stability, all the kernels that are near to each other are grouped together. The data is then segmented based on where each kernel started.

The mean shift algorithm has the following problems, [Hamerly 2003]:

- it has to find a way to group kernels and patterns, and

- as in spectral clustering, the mean shift algorithm requires the user to specify a kernel width parameter which has a profound effect on the result of the algorithm.

4. Clustering Validation Indices

The *cluster validation* problem is defined as the problem of determining the number of clusters in a data set [Langan *et al*. 1998]. The main objective of cluster validation is to evaluate clustering results in order to find the best partitiong of a data set [Halkidi *et al*. 2001]. Hence, cluster validity approaches are used to quantitatively evaluate the result of a clustering algorithm [Halkidi *et al*. 2001]. These approaches have representative indices, called *validity indices*. The traditional approach to determine the "optimum" number of clusters is to run the algorithm repetitively using different input values and to select the partitioning of data resulting in the best validity measure [Halkidi and Vazirgiannis 2001].

Two criteria that have been widely considered sufficient in measuring the quality of data partitioning, are [Halkidi *et al*. 2001]

- *Compactness*: patterns in one cluster should be similar to each other and different from patterns in other clusters. The variance of patterns in a cluster gives an indication of compactness.
- *Separation*: clusters should be well-separated from each other. The Euclidean distance between cluster centroids gives an indication of cluster separation.

There are several validity indices; a thorough survey of validity indices can be found in Halkidi *et al*. [2001]. In the following, some representative indices are discussed.

Dunn [1974] proposed a well known cluster validity index that identifies compact and well separated clusters. The main goal of Dunn's index is to maximize inter-cluster distances (i.e. separation) while minimizing intra-cluster distances (i.e. increase compactness). The Dunn index is defined as

$$D = \min_{k=1,\ldots,K} \left\{ \min_{kk=k+1,\ldots,K} \left(\frac{dist(\boldsymbol{C}_k, \boldsymbol{C}_{kk})}{\max_{a=1,\ldots,K} diam(\boldsymbol{C}_a)} \right) \right\} \quad (18)$$

where $dist(\boldsymbol{C}_k, \boldsymbol{C}_{kk})$ is the dissimilarity function between two clusters $\boldsymbol{C}_k$ and $\boldsymbol{C}_{kk}$ defined as

$$dist(\boldsymbol{C}_k, \boldsymbol{C}_{kk}) = \min_{\boldsymbol{u} \in \boldsymbol{C}_k, \boldsymbol{w} \in \boldsymbol{C}_{kk}} d(\boldsymbol{u}, \boldsymbol{w}),$$

where $d(\boldsymbol{u}, \boldsymbol{w})$ is the Euclidean distance between $\boldsymbol{u}$ and $\boldsymbol{v}$; $diam(\boldsymbol{C})$ is the diameter of a cluster, defined as

$$diam(\boldsymbol{C}) = \max_{\boldsymbol{u},\boldsymbol{w}\in\boldsymbol{C}} d(\boldsymbol{u},\boldsymbol{w})$$

An "optimal" value of K is the one that maximizes the Dunn's index. Dunn's index suffers from the following problems [Halkidi *et al*. 2001]:

- it is computationally expensive, and
- it is sensitive to the presence of noise.

Several Dunn-like indices were proposed in Pal and Biswas [1997] to reduce the sensitivity to the presence of noise.

Another well known index, proposed by Davies and Bouldin [1979], minimizes the average similarity between each cluster and the one most similar to it. The Davies and Bouldin index is defined as

$$DB = \frac{1}{K}\sum_{k=1}^{K} \max_{\substack{kk=1,\ldots,K \\ k\neq kk}} \left(\frac{diam(\boldsymbol{C}_k) + diam(\boldsymbol{C}_{kk})}{dist(\boldsymbol{C}_k, \boldsymbol{C}_{kk})} \right) \tag{19}$$

An "optimal" value of K is the one that minimizes the DB index.

Recently, Turi [2001] proposed an index incorporating a multiplier function (to penalize the selection of a small number of clusters) to the ratio between intra-cluster and inter-cluster distances, with some promising results. The index is defined as

$$V = (c \times \mathrm{N}(2,1) + 1) \times \frac{\text{intra}}{\text{inter}} \tag{20}$$

where c is a user specified parameter and N(2,1) is a Gaussian distribution with mean 2 and standard deviation of 1. The "intra" term is the average of all the distances between each data point and its cluster centroid, defined as

$$\text{intra} = \frac{1}{N_p}\sum_{k=1}^{K}\sum_{\forall \boldsymbol{u}\in\boldsymbol{C}_k} \left\| \boldsymbol{u} - \boldsymbol{m}_k \right\|^2$$

This term is used to measure the compactness of the clusters. The "inter" term is the minimum distance between the cluster centroids, defined as

$$\text{inter} = min\{\left\| \boldsymbol{m}_k - \boldsymbol{m}_{kk} \right\|^2\}, \forall\, k = 1,\ldots,K-1 \text{ and } kk = k+1,\ldots,K.$$

This term is used to measure the separation of the clusters. An "optimal" value of K is the one that minimizes the V index.

According to Turi [2001], this index performed better than both Dunn's index and the index of Davies and Bouldin on the tested cases.

Two recent validity indices are *S_Dbw* [Halkidi and Vazirgiannis 2001] and *CDbw* [Halkidi and Vazirgiannis 2002]. *S_Dbw* measures the compactness of a data set by the cluster variance, whereas separation is measured by the density between clusters. The *S_Dbw* index is defined as

$$S\_Dbw = scat(K) + Dens\_bw(K) \tag{21}$$

The first term is the average scattering of the clusters which is a measure of compactness of the clusters, defined as

$$scat(K) = \frac{1}{K}\sum_{k=1}^{K}\|\sigma(\boldsymbol{C}_k)\|/\|\sigma(\boldsymbol{Z})\|$$

where $\sigma(\boldsymbol{C}_k)$ is the variance of cluster $\boldsymbol{C}_k$ and $\sigma(\boldsymbol{Z})$ is the variance of data set $\boldsymbol{Z}$; $\|\boldsymbol{z}\|$ is defined as $\|\boldsymbol{z}\| = (\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z})^{1/2}$, where $\boldsymbol{z}$ is a vector.

The second term in Eq. (21) evaluates the density of the area between the two clusters in relation to the density of the two clusters. Thus, the second term is a measure of the separation of the clusters, defined as

$$Dens\_bw(K) = \frac{1}{K(K-1)}\sum_{k=1}^{K}\left[\sum_{\substack{k\,k=1\\ k\neq k\,k}}^{K}\frac{density(\boldsymbol{b}_{k,kk})}{max\{density(\boldsymbol{C}_k), density(\boldsymbol{C}_{kk})\}}\right]$$

where $\boldsymbol{b}_{k,kk}$ is the middle point of the line segment defined by $\boldsymbol{m}_k$ and $\boldsymbol{m}_{kk}$. The term $density(\boldsymbol{b})$ is defined as

$$density(\boldsymbol{b}) = \sum_{ll=1}^{n_{k,kk}} f(\boldsymbol{z}_{ll}, \boldsymbol{b})$$

where $n_{k,kk}$ is the total number of patterns in clusters $\boldsymbol{C}_k$ and $\boldsymbol{C}_{kk}$ (i.e. $n_{k,kk} = n_k + n_{kk}$). The function $f(\boldsymbol{z},\boldsymbol{b})$ is defined as

$$f(\boldsymbol{z},\boldsymbol{b}) = \begin{cases} 0 & \text{if } d(\boldsymbol{z},\boldsymbol{b}) > \sigma \\ 1 & \text{otherwise} \end{cases}$$

where

$$\sigma = \frac{1}{K}\sqrt{\sum_{k=1}^{K}\|\sigma(\boldsymbol{C}_k)\|}$$

An "optimal" value of K is the one that minimizes the *S_Dbw* index. Halkidi and Vazirgiannis [2001] showed that, in tested cases, *S_Dbw* successfully found the "optimal" number of clusters whereas other well-known indices often failed to do so. However, *S_Dbw* does not work properly for arbitrary shaped clusters.

To address this problem, Halkidi and Vazirgiannis [2002] proposed a multi-representative validity index, *CDbw*, in which each cluster is represented by a user-specified number of points, instead of one representative as is done in *S_Dbw*. Furthermore, *CDbw* uses intra-cluster density to measure the compactness of a data set, and uses the density between clusters to measure their separation.

More recently, Veenman *et al*. [2002; 2003] proposed a validity index that minimizes the intra-cluster variability while constraining the intra-cluster variability of the union of the two clusters. The sum of squared error is used to minimize the

intra-cluster variability while a minimum variance for the union of two clusters is used to implement the joint intra-cluster variability. The index is defined as

$$IV = min \sum_{k=1}^{K} n_k Var(\boldsymbol{C}_k) \quad (22)$$

where n_k is the number of patterns in cluster $\boldsymbol{C}_k$ and

$$Var(\boldsymbol{C}_k) = \frac{1}{n_k} \sum_{z_p \in \boldsymbol{C}_k} \left\| \boldsymbol{z}_p - \boldsymbol{m}_k \right\|^2$$

such that

$$Var(\boldsymbol{C}_k \cup \boldsymbol{C}_{kk}) \geq \sigma^2_{\max}, \quad \forall \boldsymbol{C}_k, \boldsymbol{C}_{kk}, k \neq kk$$

where $\sigma^2_{\max}$ is a user-specified parameter. This parameter has a profound effect on the final result.

The above validity indices are suitable for hard clustering. Validity indices have been developed for fuzzy clustering. The interested reader is referred to Halkidi *et al*. [2001] for more information.

These are also several information-theoretic criteria to determine the number of clusters in a data set such as Akaike's information criterion (AIC) [Akaike 1974], minimum description length (MDL) [Rissanen 1978], Merhav-Gutman-Ziv (MGZ) [Merhav 1989]. These criteria are based on likelihood and they differ in the penalty term they use to penalize large number of clusters. According to Langan *et al*. [1998], MGZ requires the user to specify *a priori* value for a parameter that has a profound effect on the resultant number of clusters. Furthermore, the penalty terms of AIC and MDL are generally useless due to the fact that the associated log likelihood function generally dominates the penalty terms in both AIC and MDL. To address this issue, Langan *et al*. [1998] proposed a cluster validation criterion that has no penalty term and applied it to the image segmentation problem with promising results.

5. Determining the Number of Clusters

Most clustering algorithms require the number of clusters to be specified in advance [Lee and Antonsson 2000; Hamerly and Elkan 2003]. Finding the "optimum" number of clusters in a data set is usually a challenge since it requires *a priori* knowledge, and/or ground truth about the data, which is not always available. The problem of finding the optimum number of clusters in a data set has been the subject of several research efforts [Halkidi *et al*. 2001; Theodoridis and Koutroubas 1999], however, despite the amount of research in this area, the outcome is still unsatisfactory [Rosenberger and Chehdi 2000]. In the literature, many approaches to dynamically find the number of clusters in a data set were proposed. In this section, several dynamic clustering approaches are presented and discussed.

ISODATA (Iterative Self-Organizing Data Analysis Technique), proposed by Ball and Hall [1967], is an enhancement of the K-means algorithm (K-means is sometimes referred to as *basic* ISODATA [Turi 2001]). ISODATA is an iterative procedure that

assigns each pattern to its closest centroids (as in K-means). However, ISODATA has the ability to merge two clusters if the distance between their centroids is below a user-specified threshold. Furthermore, ISODATA can split elongated clusters into two clusters based on another user-specified threshold. Hence, a major advantage of ISODATA compared to K-means is the ability to determine the number of clusters in a data set. However, ISODATA requires the user to specify the values of several parameters (e.g. the merging and splitting thresholds). These parameters have a profound effect on the performance of ISODATA making the result subjective [Turi 2001].

Dynamic Optimal Cluster-seek (DYNOC) [Tou 1979] is a dynamic clustering algorithm which is similar to ISODATA. DYNOC maximizes the ratio of the minimum inter-cluster distance to the maximum intra-cluster distance. This is done by an iterative procedure with the added capability of splitting and merging. However, as in ISODATA, DYNOC requires the user to specify a value for a parameter that determines whether splitting is needed [Turi 2001].

Snob [Wallace 1984; Wallace and Dowe 1994] uses various methods to assign objects to clusters in an intelligent manner [Turi 2001]. After each assignment, a means of model selection called the Wallace Information Measure (also known as the Minimum Message Length (MML)) [Wallace and Boulton 1968; Oliver and Hand 1994] is calculated and based on this calculation the assignment is accepted or rejected. Snob can split/merge and move points between clusters, thereby allowing it to determine the number of clusters in a data set.

Oliver *et al*. [1996] compares MML with different model selection methods for determining the number of clusters, K, in a data set. All the compared methods use a two step procedure where the EM algorithm is first used to estimate the parameters of each cluster for a range of K values. Then, the value of K that optimizes a tested model selection criterion (e.g. MML) is chosen. According to Oliver *et al*. [1996], MML performs better than the other examined model selection criteria when applied to the tested data sets. However, model selection methods based on the EM algorithm depend on the initial conditions and suffer from the local maximum of log-likelihood [Dai and Ma 2004].

Bischof *et al*. [1999] proposed an algorithm based on K-means which uses MDL (conceptually similar to MML). The algorithm starts with a large value for K and proceeds to remove centroids when this removal results in a reduction of the description length. K-means is used between the steps that reduce K.

Roberts *et al*. [1998] proposed a Bayesian-based approach to determine the number of clusters in a data set. The proposed approach was compared against other optimal model selection methods (including MML and MDL) on synthetic and real data sets. According to Roberts *et al*. [1998], The Bayesian methods, MDL and MML outperformed other heuristic techniques (e.g. the method proposed by Gath and Geva [1989] – discussed later in this section).

Recently, Figueiredo and Jain [2002] proposed an approach that integrates estimation and model selection in one algorithm. According to Figueiredo and Jain [2002], the proposed approach can determine the number of clusters in a data set and compared to the EM algorithm, it is less sensitive to initialization. Dai and Ma [2004] proposed a Bayesian-based approach to automatically determine the number of clusters in a data set with promising results. Furthermore, Zivkovic and van der

Heijden [2004] proposed a recursive method that estimates the parameters of the mixture and determines the number of clusters in the data set. However, the proposed approach requires the user to specify the value of a parameter, which has a profound effect on the resultant number of clusters.

Modified Linde-Buzo-Gray (MLBG), proposed by Rosenberger and Chehdi [2000], improves K-means by automatically finding the number of clusters in data set by using intermediate results. MLBG is an iterative procedure that starts with K clusters. In each iteration, a cluster, $\boldsymbol{C}_k$, maximizing an intra-cluster distance measure is chosen for splitting. Two centroids are generated from the splitting process. The first centroid, $\boldsymbol{m}_1$, is initialized to the centroid of the original cluster, $\boldsymbol{C}_k$. The second cluster centroid, $\boldsymbol{m}_2$, is chosen to be the pattern in $\boldsymbol{C}_k$ which is the most distant from $\boldsymbol{m}_1$. K-means is then applied on the new K+1 centroids. The new set of centroids is accepted if it satisfies an evaluation criterion based on a dispersion measure. This process is repeated until no valid partition of the data can be obtained. One of the main problems with MLBG is that it requires the user to specify the values of four parameters, which have a profound effect on the resultant number of clusters.

Pelleg and Moore [2000] proposed another K-means based algorithm, called X-means that uses model selection. X-means starts by setting the number of clusters, K, to be the minimum number of clusters in the data set (e.g. $K = 1$). Then, K-means is applied on the K clusters. This is followed by a splitting process based on the Bayesian Information Criterion (BIC) [Kass and Wasserman 1995] defined as

$$BIC(\boldsymbol{C} \mid \boldsymbol{Z}) = \hat{l}(\boldsymbol{Z} \mid \boldsymbol{C}) - \frac{K(N_d + 1)}{2} \log N_p \tag{23}$$

where $\hat{l}(\boldsymbol{Z} \mid \boldsymbol{C})$ is the log-likelihood of the data set $\boldsymbol{Z}$ according to model $\boldsymbol{C}$. If the splitting process improves the BIC score the resulting split is accepted, otherwise it is rejected. Other scoring functions can also be used.

These two steps are repeated until a user-specified upper bound of K is reached. X-means searches over the range of values of K and reports the value with the best BIC score.

Recently, Huang [2002] proposed SYNERACT as an alternative approach to ISODATA. SYNERACT combines K-means with hierarchical descending approaches to overcome the drawbacks of K-means mentioned previously. Three concepts used by SYNERACT are:

- a hyperplane to split up a cluster into two smaller clusters and compute their centroids,
- iterative clustering to assign pixels into available clusters, and
- a binary tree to store clusters generated from the splitting process.

According to Huang [2002], SYNERACT is faster than and almost as accurate as ISODATA. Furthermore, it does not require the number of clusters and initial location of centroids to be specified in advance. However, SYNERACT requires the user to specify the values of two parameters that affect the splitting process.

Veenman *et al*. [2002] proposed a partitional clustering algorithm that finds the number of clusters in a data set by minimizing the clustering validity index defined in Eq. (22). This algorithm starts by initializing the number of clusters equal to the number of patterns in the data set. Then, iteratively, the clusters are split or merged according to a series of tests based on the validity index. According to Veenman *et al*. [2002], the proposed approach performed better than both K-means and EM algorithms. However, the approach suffers from the following drawbacks, namely

- it is computationally expensive, and
- it requires the user to specify a parameter for the validity index (already discussed in Section 4) which has a significant effect on the final results (although the authors provide a method to help the user in finding a good value for this parameter).

More recently, Hamerly and Elkan [2003] proposed another approach based on K-means, called G-means. G-means starts with a small value for K, and with each iteration splits up the clusters whose data do not fit a Gaussian distribution. Between each round of splitting, K-means is applied to the entire data set in order to refine the current solution. According to Hamerly and Elkan [2003], G-means works better than X-means, however, it works only for data having spherical and/or elliptical clusters. G-means is not designed to work for arbitrary-shaped clusters [Hamerly 2003].

Gath and Geva [1989] proposed an unsupervised fuzzy clustering algorithm based on a combination of FCM and fuzzy maximum likelihood estimation. The algorithm starts by initializing K to a user-specified lower bound of the number of clusters in the data set (e.g. $K = 1$). A modified FCM (that uses an unsupervised learning process to initialize the K centroids) is first applied to cluster the data. Using the resulting centroids, a fuzzy maximum likelihood estimation algorithm is then applied. The fuzzy maximum likelihood estimation algorithm uses an "exponential" distance measure based on maximum likelihood estimation [Bezdek 1981] instead of the Euclidean distance measure, because the exponential distance measure is more suitable for hyper-ellipsoidal clusters. The quality of the resulting clusters is then evaluated using a clustering validity index that is mainly based on a hyper-volume criterion which measures the compactness of a cluster. K is then incremented and the algorithm is repeated until a user-specified upper bound of K is reached. The value of K resulting in the best value of the validity index is considered to be the "optimal" number of clusters in the data set. Gath and Geva [1989] stated that their algorithm works well in cases of large variability of cluster shapes. However, the algorithm becomes more sensitive to local optima as the complexity increases. Furthermore, because of the exponential function, floating point overflows may occur [Su 2002].

Lorette *et al.* [2000] proposed an algorithm based on fuzzy clustering to dynamically determine the number of clusters in a data set. A new objective function was proposed for this purpose, defined as

$$J_{\text{UFC}} = \sum_{k=1}^{K}\sum_{p=1}^{N_p} u_{k,p}^{q} d^2(\boldsymbol{z}_p, \boldsymbol{m}_k) - \beta \sum_{k=1}^{K} p_k \log(p_k) \tag{24}$$

where q is the fuzziness exponent, $u_{k,p}$ is the membership value for the p^{th} pattern in the k^{th} cluster, β is a parameter that decreases as the run progresses, and p_k is the *a priori* probability of cluster $\boldsymbol{C}_k$ defined as

$$p_k = \frac{1}{N_p} \sum_{p=1}^{N_p} u_{k,p} \quad (25)$$

The first term of Eq. (24) is the objective function of FCM which is minimized when each cluster consists of one pattern. The second term is an entropy term that is minimized when all the patterns are assigned to one cluster. Lorette *et al.* [2000] use this objective function to derive new update equations for the membership and centroid parameters.

The algorithm starts with a large number of clusters. Then, the membership values and centroids are updated using the new update equations. This is followed by applying Eq. (25) to update the *a priori* probabilities. If $p_k < \varepsilon$ then cluster k is discarded; ε is a user-specified parameter. This procedure is repeated until convergence. The drawback of this approach is that it requires the parameter ε to be specified in advance. The performance of the algorithm is sensitive to the value of ε.

Similarly, Boujemaa [2000] proposed an algorithm, based on a generalization of the competitive agglomeration clustering algorithm introduced by Frigui and Krishnapuram [1997].

The fuzzy algorithms discussed above modify the objective function of FCM. In general, these approaches are sensitive to initialization and other parameters [Frigui and Krishnapuram 1999]. Frigui and Krishnapuram [1999] proposed a robust competitive clustering algorithm based on the process of competitive agglomeration. The algorithm starts with a large number of small clusters. Then, during the execution of the algorithm, adjacent clusters compete for patterns. Clusters losing the competition will eventually disappear [Frigui and Krishnapuram 1999]. However, this algorithm also requires the user to specify a parameter that has a significant effect on the generated result.

6. Clustering using Self-Organizing Maps

Kohonen's Self Organizing Maps (SOM) [Kohonen 1995] can be used to automatically find the number of clusters in a data set. The objective of SOM is to find regularities in a data set without any external supervision [Pandya and Macy 1996]. SOM is a single-layered unsupervised artificial neural network where input patterns are associated with output nodes via weights that are iteratively modified until a stopping criterion is met [Jain *et al*. 1999]. SOM combines competitive learning (in which different nodes in the Kohonen network compete to be the winner when an input pattern is presented) with a topological structuring of nodes, such that adjacent nodes tend to have similar weight vectors (this is done via lateral feedback) [Mehrotra *et al*. 1997; Pandya and Macy 1996]. A general pseudo-code of SOM [Pandya and Macy 1996] is shown in Figure 1.

Let $\eta(t)$ be the learning rate parameter and $\Delta_w(t)$ be the neighborhood function
Randomly initialize the weight vectors, $w_k(0)$
Initialize the learning rate $\eta(0)$ and the neighborhood function $\Delta_w(0)$

Repeat
  For each input pattern z_p **do**
    Select the node whose weight vector is closest (in terms of Euclidean distance) to z_p as the winning node

    Use competitive learning to train the weight vectors such that all the nodes within the neighborhood of the winning node are moved toward z_p:

$$\boldsymbol{w}_k(t+1) = \begin{cases} \boldsymbol{w}_k(t) + \eta(t)[\boldsymbol{z}_p - \boldsymbol{w}_k(t)] & k \in \Delta_w(t) \\ \boldsymbol{w}_k(t) & \text{otherwise} \end{cases}$$

  Endloop
  Linearly decrease $\eta(t)$ and reduce $\Delta_w(t)$
Until some convergence criteria are satisfied

Figure 1. General pseudo-code for SOM

In Figure 1, $\eta(t)$ starts relatively large (e.g. close to 1) then linearly decreases until it reaches a small user-specified value. The neighborhood function $\Delta_w(t)$ defines the neighborhood size surrounding the winning node. A large value of $\Delta_w(t)$ is used at the beginning of the training. This value is then reduced as the training progresses in order to get sharper clusters [Pandya and Macy 1996]. A typical neighborhood arrangement is the rectangular lattice shown in Figure 2 [Pandya and Macy 1996].

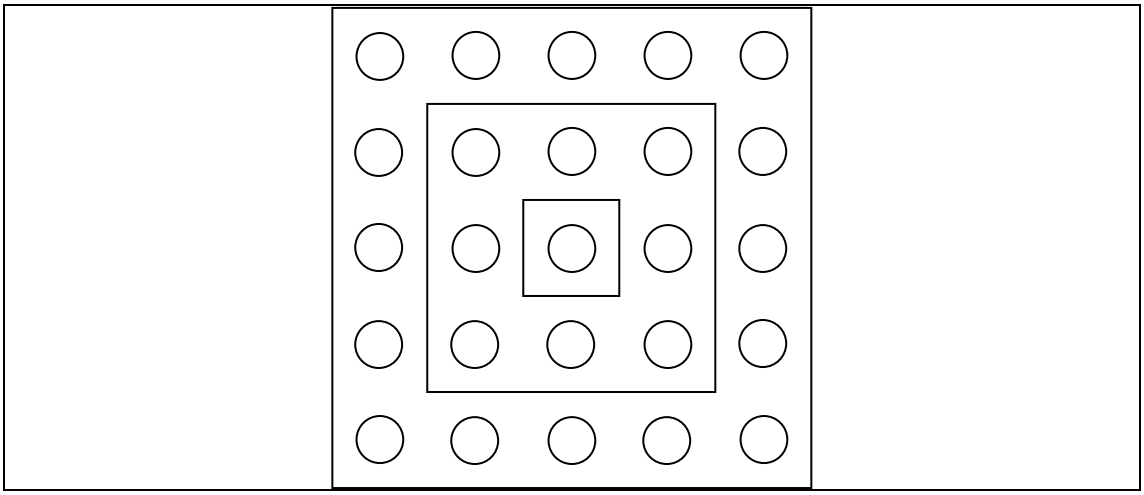

Figure 2. Rectangular Lattice arrangement of neighborhoods

SOM suffers from the following drawbacks [Jain *et al*. 1999]:

- It depends on the initial conditions.
- Its performance is affected by the learning rate parameter and the neighborhood function.
- It works well with hyper-spherical clusters only.
- It uses a fixed number of output nodes.
- It depends on the order in which the data points are presented. To overcome this problem, the choice of data points can be randomized during each iteration [Pandya and Macy 1996].

7. Clustering using Stochastic Algorithms

Simulated annealing [Van Laarhoven and Aarts 1987] has been used for clustering [Klein and Dubes 1989]. In general, a simulated annealing based clustering algorithm works as shown in Figure 3 [Jain *et al*. 1999].

An initial partition P_0 of the data set is randomly chosen
Repeat
  A neighbor of P_0 is chosen
  If the new partition is better than P_0 **then**
    move to the new partition
  else
    move to the new partition with a probability that decreases as the algorithm progresses.
Until a stopping criterion is satisfied

Figure 3. General simulated annealing based clustering algorithm

One problem with simulated annealing is that it is very slow in finding an optimal solution [Jain *et al*. 1999].

Tabu search [Glover 1989; Glover 1990] has also been used for hard clustering [Al-Sultan 1995] and fuzzy clustering [Delgado *et al*. 1997] with encouraging results. A hybrid approach combining both K-means and tabu search that performs better than both K-means and tabu search was proposed by Frnti *et al*. [1998]. Recently, Chu and Roddick [2003] proposed a hybrid approach combining both tabu search and simulated annealing that outperforms the hybrid proposed by Frnti *et al*. [1998]. However, the performance of simulated annealing and tabu search depends on the selection of several control parameters [Jain *et al*. 1999].

Most clustering approaches discussed so far perform local search to find a solution to a clustering problem. Evolutionary algorithms [Michalewicz and Fogel 2000] which perform global search have also been used for clustering [Jain *et al*. 1999]. Raghavan and Birchand [1979] used GAs [Goldberg 1989] to minimize the squared error of a clustering solution. In this approach, each chromosome represents a partition of N_p patterns into K clusters. Hence, the size of each chromosome is N_p. This representation has a major drawback in that it increases the search space by a factor of $K!$. The crossover operator may also result in inferior offspring [Jain *et al*. 1999].

Babu and Murty [1993] proposed a hybrid approach combining K-means and GAs that performed better than the GA. In this approach, a GA is only used to feed K-means with good initial centroids [Jain *et al*. 1999].

Recently, Maulik and Bandyopadhyay [2000] proposed a GA-based clustering where each chromosome represents K centroids. Hence, a floating point representation is used. The fitness function is defined as the inverse of the objective function of K-means (refer to Eq. (6)). The GA-based clustering algorithm is summarized in Figure 4.

According to Maulik and Bandyopadhyay [2000], this approach outperformed K-means on the tested cases. One drawback of this approach is that it requires the user to specify the number of clusters in advance.

1. Initialize each chromosome to contain K randomly chosen centroids from the data set
2. For $t = 1$ to $t_{\max}$

(a) For each chromosome i
  (i) Assign each pattern to the cluster with the closest centroid
  (ii) Recalculate the K cluster centroids of chromosome i as the means of their patterns
  (iii) Calculate the fitness of chromosome i

(b) Apply roulette wheel selection
(c) Apply single point crossover with probability p_c
(d) Apply mutation with probability p_m. The mutation operator is defined as

$$\boldsymbol{x} = \boldsymbol{x} \pm (r+\gamma)\boldsymbol{x}$$

where $r \sim U(0,1)$ and γ is a user-specified parameter such that $\gamma \in (0,1)$

Figure 4. General pseudo-code for GA-based clustering algorithm

Lee and Antonsson [2000] used an evolution strategy (ES) [Bäck *et al*. 1991] to dynamically cluster a data set. The proposed ES implemented variable length individuals to search for both the centroids and the number of clusters. Each individual represents a set of centroids. The length of each individual is randomly chosen from a user-specified range of cluster numbers. The centroids of each individual are then randomly initialized. Mutation is applied to the individuals by adding/subtracting a Gaussian random variable with zero mean and unit standard deviation. Two point crossover is also used as a "length changing operator". A (10+60) ES selection is used where 10 is the number of parents and 60 is the number of offspring generated in each generation. The best ten individuals from the set of parents and offspring are used for the next generation. A modification of the mean square error is used as the fitness function, defined as

$$J_{\text{ES}} = \sqrt{K+1}\sum_{k=1}^{K}\sum_{\forall \boldsymbol{z}_p \in C_k} d(\boldsymbol{z}_p, \boldsymbol{m}_k) \tag{26}$$

The modification occurs by multiplying the mean square error by a constant corresponding to the square root of the number of clusters. This constant is used to

penalize a large value of K. According to Lee and Antonsson [2000], the results are promising. However, the proposed algorithm needs to be compared with other dynamic clustering approaches and its performance needs to be investigated as the dimension increases.

In general, evolutionary approaches have several advantages, namely [Jain *et al*. 1999]:

- they are global search approaches,
- they are suitable for parallel processing, and
- they can work with a discontinuous criterion function.

However, evolutionary approaches generally suffer from the following drawbacks [Jain *et al*. 1999]:

- they require the user to specify the values of a set of parameters (e.g. population size, p_c, p_m, etc.) for each specific problem, and
- the execution time of EAs is significantly higher than the execution time of other traditional clustering algorithms (e.g. K-means and FCM), especially when applied to large data sets.

More recently, Omran *et al*. [2002; 2005] proposed a Particle Swarm Optimization (PSO) [Kennedy and Eberhart 1995]-based clustering algorithm where each particle represents K centroids. Hence, a floating point representation is used. According to Omran *et al*. [2005], this approach generally outperformed K-means, FCM, KHM, H2 and GA on the tested cases. One drawback of this approach is that it requires the user to specify the number of clusters in advance.

To address this drawback, a dynamic clustering approach based on PSO, was proposed by Omran [2005]. The proposed approach automatically determines the "optimum" number of clusters and simultaneously clusters the data set with minimal user interference. The algorithm starts by partitioning the data set into a relatively large number of clusters to reduce the effects of initial conditions. Using binary PSO the "best" number of clusters is selected. The centroids of the chosen clusters are then refined via the K-means clustering algorithm. The experiments conducted by Omran [2005] show that the proposed approach generally found the "optimum" number of clusters on the tested cases.

Recently, Differential Evolution [Storn and Price 1995] was applied to the clustering problem by Paterlini and Krink [2004] and Omran *et al*. [2005] with promising results.

8. Summary

This paper presented an overview of the different clustering methods. First the data clustering problem was defined. This was followed by defining the terms used in this paper. In addition, a brief overview of the different similarity measures was given. Clustering techniques were then discussed. A presentation of different clustering validation techniques was then shown. Methods that automatically determine the number of clusters in a data set was then presented. Finally, an overview of clustering using SOMs and stochastic techniques was presented.

References

H. Abbas and M. Fahmy. Neural Networks for Maximum Likelihood Clustering. *Signal Processing*, vol. 36, no.1, pp. 111-126, 1994.

H. Akaike. A New Look at the Statistical Model Identification. *IEEE Transactions on Automated Control*, vol. AC-19, Dec., 1974.

N. Alldrin, A. Smith and D. Turnbull. Clustering with EM and K-means, unpublished Manuscript, 2003, http://louis.ucsd.edu/~nalldrin/research/cse253_wi03.pdf (visited 15 Nov 2003).

K. Al-Sultan. A Tabu Search Approach to Clustering Problems. *Pattern Recognition*, vol. 28, pp. 1443-1451, 1995.

M. Anderberg. *Cluster Analysis for Applications*. Academic Press, New York, USA, 1973.

G. Babu and M. Murty. A Near-Optimal Initial Seed Value Selection in K-means Algorithm Using a Genetic Algorithm. *Pattern Recognition Letters*, vol. 14, no. 10, pp. 763-769, 1993.

F. Bach and M. Jordan. Learning Spectral Clustering. *Neural Information Processing Systems 16 (NIPS 2003)*, 2003.

T. Bäck, F. Hoffmeister and H. Schwefel. A Survey of Evolution Strategies. In *Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications*, pp. 2-9, 1991.

S. Baek, B. Jeon, D. Lee and K. Sung. Fast Clustering Algorithm for Vector Quantization. *Electronics Letters*, vol. 34, no. 2, pp. 151-152, 1998.

G. Ball and D. Hall. A Clustering Technique for Summarizing Multivariate Data. *Behavioral Science*, vol. 12, pp. 153-155, 1967.

J. Bezdek. A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, pp. 1-8, 1980.

J. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, 1981.

H. Bischof, A. Leonardis and A. Selb. MDL Principle for Robust Vector Quantization. *Pattern Analysis and Applications*, vol. 2, pp. 59-72, 1999.

C. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

N. Boujemaa. On Competitive Unsupervised Clustering. In the International Conference on Pattern Recognition (ICPR'00), vol. 1, pp. 1631-1634, 2000.

C. Carpineto and G. Romano. A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval. *Machine Learning*, vol. 24, no. 2, pp. 95-122, 1996.

S. Chu and J. Roddick. A Clustering Algorithm Using Tabu Search Approach with Simulated Annealing for Vector Quantization. *Chinese Journal of Electronics*, vol. 12**,** no. 3, pp. 349-353, 2003.

F. Chung. *Spectral Graph Theory*. Society Press, 1997.

G. Coleman and H. Andrews. Image Segmentation by Clustering. In *Proceedings of IEEE*, vol. 67, pp. 773-785, 1979.

D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603-619, 2002.

H. Dai and W. Ma. A Novelty Bayesian Method for Unsupervised Learning of Finite Mixture Models. In *Proceedings of the 3rd International Conference on Machine Learning and Cybernetics*, Shanghai, China, pp. 3574-3578, 2004.

E. Davies. *Machine Vision: Theory, Algorithms, Practicalities*. Academic Press, 2nd Edition, 1997.

D. Davies and D. Bouldin. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, 1979.

M. Delgado, A. Skarmeta and H. Barberá. A Tabu Search Approach to the Fuzzy Clustering Problem. In *the Sixth IEEE International Conference on Fuzzy Systems*, Barcelona, 1997.

J. C. Dunn. Well Separated Clusters and Optimal Fuzzy Partitions. *Journal of Cybernetics*, vol. 4, pp. 95-104, 1974.

B. Everitt. *Cluster Analysis*. Heinemann Books, London, 1974.

M. Figueiredo and A. Jain. Unsupervised Learning of Finite Mixture Models. IEEE Transactions on Pattern Analysis and Machine Intillegence, vol. 24, no. 3, pp. 381-396, 2002.

E. Forgy. Cluster Analysis of Multivariate Data: Efficiency versus Interpretability of Classification. *Biometrics*, vol. 21, pp. 768-769, 1965.

H. Frigui and R. Krishnapuram. Clustering by Competitive Agglomeration. *Pattern Recognition Letters*, vol. 30, no. 7, pp. 1109-1119, 1997.

H. Frigui and R. Krishnapuram. A Robust Competitive Clustering Algorithm with Applications in Computer Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no.5, pp. 450-465, 1999.

P. Frnti, J. Kivijrvi and O. Nevalainen. Tabu Search Algorithm for Codebook Generation in Vector Quantization. *Pattern Recognition*, vol. 31, no. 8, pp. 1139-1148, 1998.

I. Gath and A. Geva. Unsupervised Optimal Fuzzy Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 773-781, 1989.

F. Glover. Tabu Search – Part I. *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190-206, 1989.

F. Glover. Tabu Search – Part II. *ORSA Journal on Computing*, vol. 2, no. 1, pp. 4-32, 1990.

D. Goldberg. *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

M. Halkidi, Y. Batistakis and M. Vazirgiannis. On Clustering Validation Techniques. *Intelligent Information Systems Journal*, Kluwer Pulishers, vol. 17, no. 2-3, pp.107-145, 2001.

M. Halkidi and M. Vazirgiannis. Clustering Validity Assessment: Finding the Optimal Partitioning of a data set. In *Proceedings of ICDM Conference*, CA, USA, 2001.

M. Halkidi and M. Vazirgiannis. Clustering Validity Assessment using Multi representative. In *Proceedings of the Hellenic Conference on Artificial Intelligence, SETN*, Thessaloniki, Greece, 2002.

G. Hamerly. Learning Structure and Concepts in Data using Data Clustering, *PhD Thesis*. University of California, San Diego, 2003.

G. Hamerly and C. Elkan. Alternatives to the K-means Algorithm that Find Better Clusterings. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM-2002)*, pp. 600-607, 2002.

G. Hamerly and C. Elkan. Learning the K in K-means. In *The Seventh Annual Conference on Neural Information Processing Systems*, 2003.

K. Huang. A Synergistic Automatic Clustering Technique (Syneract) for Multispectral Image Analysis. *Photogrammetric Engineering and Remote Sensing*, vol. 1, no.1, pp. 33-40, 2002.

A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice Hall, New Jersey, USA, 1988.

A. Jain, R. Duin and J. Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no.1, pp. 4-37, 2000.

A. Jain, M. Murty and P. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, vol. 31, no. 3, pp. 264-323,1999.

D. Judd, P. Mckinley and A. Jain. Large-scale Parallel Data Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 871-876, 1998.

R. Kass and L. Wasserman. A Reference Bayesian Test for Nested Hypotheses and its Relationship to the Schwarz Criterion. *Journal of the American Statistical Association*, vol. 90, no. 431, pp. 928-934, 1995.

T. Kaukoranta, P. Fränti and O. Nevalainen. A New Iterative Algorithm for VQ Codebook Generation. *International Conference on Image Processing*, pp. 589-593, 1998.

J. Kennedy and R. Eberhart. Particle Swarm Optimization. In *Proceedings of IEEE International Conference on Neural Networks*, Perth, Australia, vol. 4, pp. 1942-1948, 1995.

R. Klein and R. Dubes. Experiments in Projection and Clustering by Simulated Annealing. *Pattern Recognition*, vol. 22, pp. 213-220, 1989.

T. Kohonen. *Self-Organizing Maps*. Springer Series in Information Sciences, 30, Springer-Verlag, NewYork, USA, 1995.

Krishnapuram and Keller. A Possibilistic Approach to Clustering. *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 2, pp. 98-110, 1993.

Krishnapuram and Keller. The Possibilistic C-Means algorithm: Insights and Recommendations. *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 3, pp. 385-393, 1996.

D. Langan, J. Modestino and J. Zhang. Cluster Validation for Unsupervised Stochastic Model-Based Image Segmentation. *IEEE Transactions on Image Processing*, vol. 7, no. 2, pp. 180-195, 1998.

C. Lee and E. Antonsson. Dynamic Partitional Clustering Using Evolution Strategies. In *The Third Asia-Pacific Conference on Simulated Evolution and Learning*, 2000.

Y. Leung, J. Zhang and Z. Xu. Clustering by Space-Space Filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no.12, pp. 1396-1410, 2000.

A. Liew, S. Leung and W. Lau. Fuzzy Image Clustering Incorporating Spatial Continuity. In *IEE Proceedings Vision, Image and Signal Processing*, vol. 147, no. 2, 2000.

A. Lorette, X. Descombes and J. Zerubia. Fully Unsupervised Fuzzy Clustering with Entropy Criterion. In *International Conference on Pattern Recognition (ICPR'00)*, vol. 3, pp. 3998-4001, 2000.

S. Lu and K. Fu. A Sentence-to-Sentence Clustering Procedure for Pattern Analysis. *IEEE Transaction on Systems, Man and Cybernetics*, vol. 8, pp. 381-389, 1978.

J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings Fifth Berkeley Symposium on Mathematics, Statistics and Probability*, vol. 1, pp. 281-297, 1967.

U. Maulik and S. Bandyopadhyay. Genetic Algorithm-Based Clustering Technique. *Pattern Recognition*, vol. 33, pp. 1455-1465, 2000.

G. McLachlan and T. Krishnan. *The EM algorithm and Extensions*. John Wiley & Sons, Inc., 1997.

K. Mehrotra, C. Mohan and Rakka. *Elements of Artificial Neural Networks*. MIT Press, 1997.

N. Merhav. The Estimation of the Model Order in Exponential Families. *IEEE Transactions on Information T*heory, vol. 35, Sep. 1989.

Z. Michalewicz and D. Fogel. *How to Solve It: Modern Heuristics*. Springer-Verlag, Berlin, 2000.

A. Ng, M. Jordan and Y. Weiss. On Spectral Clustering: Analysis and an Algorithm. In *Proceedings of Neural Information Processing Systems (NIPS 2001)*, 2001.

J. Oliver, R. Baxter and C. Wallace. Unsupervised Learning using MML. In *Proceedings of the 13th International Conference Machine Learning (ICML'96)*, pp. 364-372, San Francisco, USA, 1996.

J. Oliver and D. Hand. Introduction to Minimum Encoding Inference. Technical Report no. 94/205. Department of Computer Science, Monash University, Australia, 1994.

M. Omran. *Particle Swarm Optimization Methods for Pattern Recognition and Image Processing*, *PhD Thesis*. Department of Computer Science, University of Pretoria, South Africa, 2005.

M. Omran, A. Engelbrecht and A. Salman. Differential Evolution Methods for Unsupervised Image Classification. To appear in the *IEEE Congress on Evolutionary Computation (CEC2005)*, September 2005.

M. Omran, A. Engelbrecht and A. Salman. Particle Swarm Optimization Method for Image Clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, no. 3, pp. 297-322, May 2005.

M. Omran, A. Salman and A. Engelbrecht. Image Classification using Particle Swarm Optimization. In *Conference on Simulated Evolution and Learning*, Singapore, pp. 370-374, November 2002.

N. Pal and J. Biswas. Cluster Validation using Graph Theoretic Concepts. *Pattern Recognition*, vol. 30, no. 6, 1997.

A. Pandya and R. Macy. *Pattern Recognition with Neural Networks in C++*. CRC Press, 1996.

S. Paterlini and T. Krink. High Performance Clustering with Differential Evolution. In the *Congress on Evolutionary Computation (CEC2004)*, vol. 2, pp. 2004-2011, 2004.

D. Pelleg and A. Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 727-734, Morgan Kaufmann, San Francisco, CA, 2000.

V. Raghavan and K. Birchand. A Clustering Strategy Based on a Formalism of the Reproductive Process in a Natural System. In *Proceedings of the Second International Conference on Information Storage and Retrieval*, pp. 10-22, 1979.

R. Rendner and H. Walker. Mixture Densities, Maximum Likelihood and the EM Algorithm. *SIAM Review*, vol. 26, no. 2, 1984.

J. Rissanen. Modeling by Shortest Data Description. *Automatica,* vol. 14, pp. 465-471, 1978.

S. Roberts, D. Husmeier, L. Rezek and W. Penny. Bayesian Approaches to Gaussian Mixture Modeling. *IEEE Transactions in Pattern Recognition and Machine Intelligence*, vol. 20, no. 11, pp. 1133-1142, 1998.

C. Rosenberger and K. Chehdi. Unsupervised Clustering Method with Optimal Estimation of the Number of Clusters: Application to Image Segmentation. In *The International Conference on Pattern Recognition (ICPR'00)*, vol. 1, pp. 1656-1659, 2000.

P. Scheunders. A Comparison of Clustering Algorithms Applied to Color Image Quantization. *Pattern Recognition Letters*, vol. 18, no. 11-13, pp. 1379-1384, 1997.

P. Sneath and R. Sokal. *Numerical Taxonomy*. Freeman, London, UK, 1973.

R. Storn. and K. Price. Differential Evolution – a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, Technical Report TR-95-012, ICSI, 1995.

M. Su. Cluster Analysis: Chapter two Lecture notes, 2002, http://selab.csie.ncu.edu.tw/~muchun/course/cluster/CHAPTER%202.pdf (visited 15 August 2004).

S. Theodoridis and K. Koutroubas. *Pattern Recognition*. Academic Press, 1999.

J. Tou. DYNOC – A Dynamic Optimal Cluster-seeking Technique. *International Journal of Computer and Information Sciences*, vol. 8, no. 6, pp. 541-547, 1979.

R.H. Turi. *Clustering-Based Colour Image Segmentation, PhD Thesis*. Monash University, Australia, 2001.

P. Van Laarhoven and E. Aarts. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, 1987.

C. Veenman, M. Reinders and E. Backer. A Maximum Variance Cluster Algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1273-1280, 2002.

C. Veenman, M. Reinders and E. Backer. A Cellular Coevolutionary Algorithm for Image Segmentation. *IEEE Transactions on Image Processing*, vol. 12, no. 3, pp. 304-316, 2003.

C. Wallace. An Improved Program for Classification. Technical Report no. 47. Department of Computer Science, Monash University, Australia, 1984.

C. Wallace and D. Boulton. An Information Measure for Classification. *The Computer Journal*, vol. 11, pp. 185-194, 1968.

C. Wallace and D. Dowe. Intrinsic Classification by MML – the snob program. In *Proceedings Seventh Australian Joint Conference on Artificial Intelligence, UNE*, Armidale, NSW, Australia, pp. 37-44, 1994.

Z. Xiang. Color Image Quantization by Minimizing the Maximum Inter-cluster Distance. *ACM Transactions on Graphics*, vol. 16, no. 3, pp. 260-276, 1997.

B. Zhang. Generalized K-Harmonic Means - Boosting in Unsupervised Learning. Technical Report HPL-2000-137. Hewlett-Packard Labs, 2000.

B. Zhang, M. Hsu and U. Dayal. K-Harmonic Means - A Data Clustering Algorithm. Technical Report HPL-1999-124. Hewlett-Packard Labs, 1999.

Z. Zivkovic and F. van der Heijden. Recursive Unsupervised Learning of Finite Mixture Models. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 651-656, 2004.