An overview of clustering methods

Mahamed G.H. Omran[a,*], Andries P. Engelbrecht[b] and Ayed Salman[c]
[a]*Department of Computer Science, Gulf University for Science and Technology, Kuwait*
[b]*Department of Computer Science, School of Information Technology, University of Pretoria, Pretoria 0002, South Africa*
[c]*Department of Computer Engineering, Kuwait University, Kuwait*

Received 15 Septemer 2006
Revised 21 December 2006
Accepted 12 February 2007

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) [51,52]. It is an important process in pattern recognition and machine learning [47]. Furthermore, data clustering is a central process in Artificial Intelligence (AI) [46]. Clustering algorithms are used in many applications, such as image segmentation [22,50,100], vector and color image quantization [9,56,109], data mining [53], compression [1], machine learning [17], etc. A cluster is usually identified by a cluster center (or *centroid*) [64]. Data clustering is a difficult problem in unsupervised pattern recognition as the clusters in data may have different shapes and sizes [51].

Several surveys and books on clustering have been published [11,31,51,52,56]. However, due to the prohibitive amount of research conducted in the area of clustering and the number of new clustering approaches proposed during the last few years, an up-to-date survey investigating the *state-of-the-art* clustering methods is needed. 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 easy-to-read and up-to-date overview of a set of representative clustering methods. Furthermore, this paper tries to address the clustering problems from different angles by presenting classical and new clustering methods, discussing

[*]Corresponding author. E-mail: mjomran@gmail.com.

the problem of finding the "optimal" number of clusters in a data set and providing an overview of using stochastic approaches to solve clustering problems.

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 [52].
- A *feature* (or *attribute*) is an individual component of a pattern [52].
- A *cluster* is a set of similar patterns, and patterns from different clusters are not similar [30].
- *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 [52].

The clustering problem can be formally defined as follows [104]:

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.

  $\boldsymbol{C}_k \neq \phi, k = 1, \ldots, K$
- Each pattern is assigned to one and only one cluster (in case of hard clustering only), i.e.

  $\boldsymbol{C}_k \cap \boldsymbol{C}_{kk} = \phi$ 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 [52].

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

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

Euclidean distance is a special case (when $\alpha = 2$) of the Minkowski metric [52] defined as

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

When $\alpha = 1$, the measure is referred to as the Manhattan distance [48].

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 [48]. 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 [52]. 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

$$< \boldsymbol{z}_u, \boldsymbol{z}_w > = \frac{\sum\limits_{j=1}^{N_d} z_{u,j} z_{w,j}}{\|\boldsymbol{z}_u\| \; \|\boldsymbol{z}_w\|} \tag{3}$$

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 [48].

Another distance measure is the Mahalanobis distance defined as

$$d_M(\boldsymbol{z}_u, \boldsymbol{z}_w) = (\boldsymbol{z}_u - \boldsymbol{z}_w)\Sigma^{-1}(\boldsymbol{z}_u - \boldsymbol{z}_w)^T \tag{4}$$

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 [52].

There are other distance measures in the literature. Readers are referred to Everitt et al. [31] and Fielding [32] for more details.

3. Clustering techniques

Most clustering algorithms are based on two popular techniques known as *hierarchical* and *partitional* clustering [36,65]. 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 [46]. A cluster tree is defined as “a tree showing a sequence of clustering with each clustering being a partition of the data set” [65]. 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 [100]. 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* [95] and *complete link* [5] 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 [100]. 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 [52]. Another less popular agglomerative hierarchical algorithm is the *centroid* method [5]. 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 [100]. More details about traditional hierarchical clustering techniques can be found in Everitt [30].

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. [65]. The following paragraph provides the reader with a good idea about this approach as described by Leung et al. [65]:

> "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. [65], 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 [36]:

– 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 $O(N_p^2)$ [100]). 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 [65]. 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 [51], hence, this paper concentrates on partitional techniques.

Partitional clustering algorithms are generally iterative algorithms that converge to local optima [47]. Employing the general form of iterative clustering used by Hamerly and Elkan [47], 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 $u(\boldsymbol{m}_k|\boldsymbol{z}_p)$ to each centroid $\boldsymbol{m}_k$ and its weight $w(\boldsymbol{z}_p)$
   – Recalculate the $K$ cluster centroids, using
     $$\boldsymbol{m}_k = \frac{\sum_{\forall \boldsymbol{z}_p} u(\boldsymbol{m}_k|\boldsymbol{z}_p)w(\boldsymbol{z}_p)\boldsymbol{z}_p}{\sum_{\forall \boldsymbol{z}_p} u(\boldsymbol{m}_k|\boldsymbol{z}_p)w(\boldsymbol{z}_p)} \quad (5)$$
   **until** a stopping criterion is satisfied.
```

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

– $u(\boldsymbol{m}_k|\boldsymbol{z}_p) \geqslant 0$, $p = 1,\ldots, N_p$ and $k = 1,\ldots,K$
– $\sum_{k=1}^{K} u(\boldsymbol{m}_k|\boldsymbol{z}_p) = 1,\ p = 1,\ldots, N_p$

Crisp clustering algorithms use a *hard* membership function (i.e. $u(\boldsymbol{m}_k|\boldsymbol{z}_p) \in\{0,1\}$), while fuzzy clustering algorithms use a *soft* member function (i.e. $u(\boldsymbol{m}_k|\boldsymbol{z}_p) \in[0,1]$) [47].

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 $w(\boldsymbol{z}_p) > 0$ [47]. The weight function was proposed by Zhang [111].

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 [34]. The objective function that the K-means optimizes is

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

Hence, the K-means algorithm minimizes the intra-cluster distance [47]. 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

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

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

Hence, K-means has a hard membership function. Furthermore, K-means has a constant weight function, thus, all patterns have equal importance [47].

The K-means algorithm has the following main advantages [100]:

– 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 [25]:

– 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 [46]. The objective of K-medoids is to find the most centrally located patterns within the clusters [42]. 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 [12,13]. 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 $q \geqslant 1$. 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:

– $u_{k,p} \geqslant 0, p = 1, \ldots, N_p$ and $k = 1, \ldots, K$
– $\sum\limits_{k=1}^{K} u_{k,p} = 1, p = 1, \ldots, N_p$

The membership and weight functions for FCM are defined as [47]

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

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

Hence, FCM has a soft membership function and a constant weight function. In general, FCM performs better than K-means [46] and it is less affected by the presence of uncertainty in the data [66]. 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 [52].

Krishnapuram and Keller [60,61] 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 [100]. Possibilistic C-means works well in the presence of noise in the data set. However, it has several drawbacks, namely [100],

– 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.

A thorough survey on fuzzy clustering approaches can be found in Baraldi and Blonda [11].

3.2.3. The Gaussian Expectation-Maximization algorithm

Another popular clustering algorithm is the Expectation-Maximization (EM) algorithm [15,72,90]. EM is used for parameter estimation in the presence of some unknown data [46]. 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 [3]. The objective function that the EM optimizes as defined by Hamerly and Elkan [47] is

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

where $p(\boldsymbol{z}_p|\boldsymbol{m}_k)$ 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 [47]

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

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

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 [46]. 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. [103] and Hamerly [46] showed that K-means performs comparably to EM. Furthermore, Aldrin et al. [3] stated that EM fails on high-dimensional data sets due to numerical precision problems. They also observed that Gaussians often collapsed to delta functions [3]. In addition, EM depends on the initial estimate of the parameters [46,100] 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 [76].

3.2.4. The K-harmonic means algorithm

Recently, Zhang and colleagues [111,112] 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

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

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

The membership and weight functions for KHM are [47]

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

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

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 [47].

Contrary to K-means, KHM is less sensitive to initial conditions and does not have the problem of collapsing Gaussians exhibited by EM [3]. Experiments conducted by Zhang et al. [112], Zhang [111] and Hamerly and Elkan [47] showed that KHM outperformed K-means, FCM (according to Hamerly and Elkan [47]) and EM.

3.2.5. Hybrid 2

Hamerly and Elkan [47] 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 [46], 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 [70]. 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 [94]. 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 [25]. 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 [68]. 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 [52].

Recently, a new type of clustering algorithms called *spectral* clustering algorithms [7,76] has been proposed by computer vision researchers and graph theorists. Spectral clustering is based on spectral graph theory [20] 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 [46]. One major advantage of spectral clustering is that it can generate arbitrary-shaped clusters. However, spectral clustering suffers from two major drawbacks [46]:

- 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 [23] 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 [46]:

- 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 [63]. The main objective of cluster validation is to evaluate clustering results in order to find the best partitiong of a data set [42]. Hence, cluster validity approaches are used to quantitatively evaluate the result of a clustering algorithm [42]. 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 [43].

Two criteria that have been widely considered sufficient in measuring the quality of data partitioning, are [42]

- *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. [42]. In the following, some representative indices are discussed.

Dunn [29] 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

$$D = \min_{k=1,\ldots,K} \left\{ \min_{kk=k+1,\ldots,K} \left(\frac{dist(C_k, C_{kk})}{\max\limits_{a=1,\ldots,K} diam(C_a)} \right) \right\} \tag{18}$$

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

$$dist(C_k, C_{kk}) = \min_{u \in C_k, w \in C_{kk}} d(u, w),$$

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

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

An "optimal" value of K is the one that maximizes the Dunn's index. Dunn's index suffers from the following problems [42]:

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

Several Dunn-like indices were proposed in Pal and Biswas [85] to reduce the sensitivity to the presence of noise.

Another well known index, proposed by Davies and Bouldin [26], minimizes the average similarity between each cluster and the one most similar to it. The Davies and Bouldin index is defined as

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

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

Recently, Turi [100] 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

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

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

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

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

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

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 [100], 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* [43] and *CDbw* [44]. *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

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

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 $||\boldsymbol{z}|| = (\boldsymbol{z}^T\boldsymbol{z})^{1/2}$, 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

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

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

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

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 [43] 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 [44] 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. [103,104] 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

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

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

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

such that

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

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. [42] 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) [2], minimum description length (MDL) [91], Merhav-Gutman-Ziv (MGZ) [74]. 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. [63], 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. [63] 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 [46,64]. 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 [42,98], however, despite the amount of research in this area, the outcome is still unsatisfactory [93]. 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 [10], is an enhancement of the K-means algorithm (K-means is sometimes referred to as *basic* ISODATA [100]). 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 [100].

Dynamic Optimal Cluster-seek (DYNOC) [99] 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 [100].

Snob [105,107] uses various methods to assign objects to clusters in an intelligent manner [100]. After each assignment, a means of model selection called the Wallace Information Measure (also known as the Minimum Message Length (MML)) [78,106] 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. [77] 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. [77], 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 [24].

Bischof et al. [14] 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. [92] 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. [92], the Bayesian methods, MDL and MML outperformed other heuristic techniques (e.g. the method proposed by Gath and Geva [38] – discussed later in this section).

Recently, Figueiredo and Jain [33] proposed an approach that integrates estimation and model selection in one algorithm. According to Figueiredo and Jain [33], 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 [24] 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 [113] 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 [93], 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 [88] 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) [55] defined as

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

where $\hat{l}(\boldsymbol{Z}|\boldsymbol{C})$ 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 [49] 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 [49], 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. [103] 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. [103], 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 [48] 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 [48], 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 [46].

Gath and Geva [38] 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 [13] 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 [38] 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 [97].

Lorette et al. [67] 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

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

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

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

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. [67] 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 [16] proposed an algorithm, based on a generalization of the competitive agglomeration clustering algorithm introduced by Frigui and Krishnapuram [35].

```
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
```

Fig. 1. General pseudo-code for SOM.

The fuzzy algorithms discussed above modify the objective function of FCM. In general, these approaches are sensitive to initialization and other parameters [36]. Frigui and Krishnapuram [36] 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 [36]. 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) [59] 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 [86]. 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 [52]. 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) [73,86]. A general pseudo-code of SOM [86] is shown in Fig. 1.

In Fig. 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 [86]. A typical neighborhood arrangement is the rectangular lattice shown in Fig. 2 [86].

SOM suffers from the following drawbacks [52]:

– 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 [86].

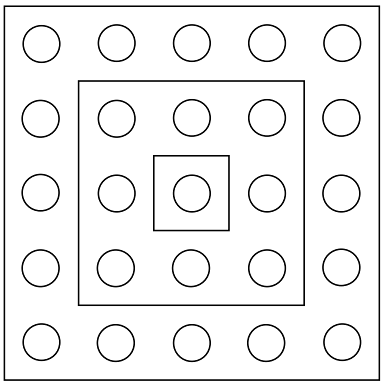


Fig. 2. Rectangular Lattice arrangement of neighborhoods.

```
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
```

Fig. 3. General simulated annealing based clustering algorithm.

7. Clustering using stochastic algorithms

Simulated annealing [102] has been used for clustering [58]. In general, a simulated annealing based clustering algorithm works as shown in Fig. 3 [52].

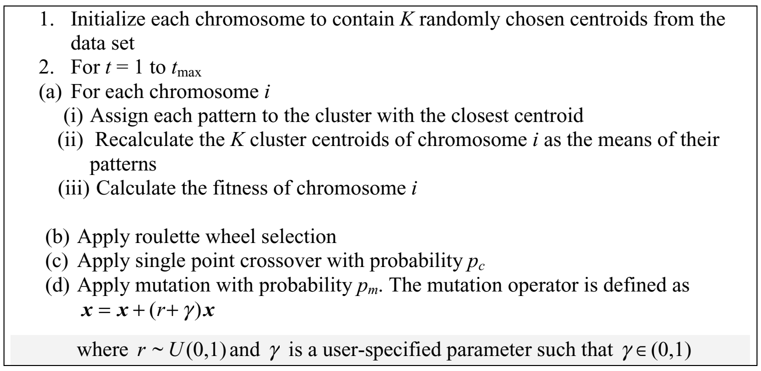
One problem with simulated annealing is that it is very slow in finding an optimal solution [52].

Tabu search [39,40] has also been used for hard clustering [4] and fuzzy clustering [27] 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. [37]. Recently, Chu and Roddick [19] proposed a hybrid approach combining both tabu search and simulated annealing that outperforms the hybrid proposed by Frnti et al. [37]. However, the performance of simulated annealing and tabu search depends on the selection of several control parameters [52].

Most clustering approaches discussed so far perform local search to find a solution to a clustering problem. Evolutionary algorithms [75] which perform global search have also been used for clustering [52]. Raghavan and Birchand [89] used GAs [41] 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 [52].

Babu and Murty [6] 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 [52].

Recently, Maulik and Bandyopadhyay [71] 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 Fig. 4.



```
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
     $\boldsymbol{x} = \boldsymbol{x} + (r + \gamma)\boldsymbol{x}$
     where $r \sim U(0,1)$ and $\gamma$ is a user-specified parameter such that $\gamma \in (0,1)$
```

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

According to Maulik and Bandyopadhyay [71], 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.

Lee and Antonsson [64] used an evolution strategy (ES) [8] 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

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

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 [64], 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 [52]:

– 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 [52]:

– 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. [81,84] proposed a Particle Swarm Optimization (PSO) [57]-based clustering algorithm where each particle represents K centroids. Hence, a floating point representation is used. According to Omran et al. [81], this approach generally outperformed K-means, FCM, KHM, H2 and GA on the tested cases. The same algorithm of Omran et al. [81,84] was used by Van der Merwe and Engelbrecht [74] to cluster general data sets. It was applied on a set of multi-dimensional data (e.g. the Iris plant data base). In general, the results show that the PSO-based clustering algorithm performs better than the K-means algorithm, which verify the results of Omran et al. [81,84].

One drawback of the above PSO-based approaches is that they require 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 [79]. 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 [79] show that the proposed approach generally found the "optimum" number of clusters on the tested cases.

Xiao et al. [110] proposed a hybrid approach to solve gene clustering problems. The proposed approach uses PSO and SOM such that SOM groups the data set and PSO optimizes the SOM's weights. The results show that the hybrid approach provides clusters with greater compactness.

Cohen and de Castro [21] proposed a PSO-based approach where each particle represents a prototype for a cluster. Thus, the particle represents only one part of the solution. In addition, the proposed approach does not use a fitness function. The particles are moved into regions in the search space representing natural clusters.

Recently, Differential Evolution [96] was applied to the clustering problem by Paterlini and Krink [87] and Omran et al. [80] with promising results. A new variant of DE was proposed by Omran et al. [83] where the control parameters are self-adaptive (i.e. the user does not need to specify the DE's control parameters *a priori*). The new variant was called *Self-adaptive Differential Evolution* (SDE). The experiments conducted show that SDE generally outperform other DE algorithms in all the benchmark functions. An SDE-based clustering algorithm was proposed by Omran and Engelbrecht [82] with encouraging results.

Clustering approaches inspired by the collective behaviors of ants have been proposed by [62,69,108]. The main idea of these approaches is that artificial ants are used to pick up items and drop them near similar items resulting in the formation of clusters.

Biclustering [18,45] is a way to cluster rows and columns of a matrix at the same time. The objective was to find minimum variance biclusters. Divina and Aguilar-Ruiz [28] proposed an evolutionary-based approach to search for biclusters with promising results.

8. Summary

Clustering is the process of finding natural groups within a data set such that patterns within a group are more similar to each other than patterns belonging to different groups. Clustering has been used in a wide range of scientific and engineering disciplines. Clustering is a difficult problem with complex mathematical modeling. In this paper, different clustering approaches were presented and discussed. These approaches differ considerably in terms of efficiency, cost, solution quality, etc. Each approach has its strengths, weaknesses and limitations. If speed is our main concern then FCM and KHM seem to be reasonable choices. However, if the quality of the solution is our main objective then population-based stochastic approaches represent viable options because they can, in general, avoid being trapped in local optima.

References

[1] H. Abbas and M. Fahmy. Neural Networks for Maximum Likelihood Clustering, *Signal Processing* **36**(1) (1994), 111–126.

[2] H. Akaike, A New Look at the Statistical Model Identification, *IEEE Transactions on Automated Control* **AC-19** (Dec. 1974).

[3] 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).

[4] K. Al-Sultan, A Tabu Search Approach to Clustering Problems, *Pattern Recognition* **28** (1995), 1443–1451.

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

[6] G. Babu and M. Murty, A Near-Optimal Initial Seed Value Selection in K-means Algorithm Using a Genetic Algorithm, *Pattern Recognition Letters* **14**(10) (1993), 763–769.

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

[8] 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*, 199, 2–9.

[9] S. Baek, B. Jeon, D. Lee and K. Sung, Fast Clustering Algorithm for Vector Quantization, *Electronics Letters* **34**(2) (1998), 151–152.

[10] G. Ball and D. Hall, A Clustering Technique for Summarizing Multivariate Data, *Behavioral Science* **12** (1967), 153–155.

[11] A. Baraldi and P. Blonda, A Survey of Fuzzy Clustering Algorithms for Pattern Recognition-Part I and II, *IEEE Transaction on Systems, Man, and Cybernetics, Part B* **29**(6) (1999), 778–801.

[12] J. Bezdek, A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2** (1980), 1–8.

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

[14] H. Bischof, A. Leonardis and A. Selb, MDL Principle for Robust Vector Quantization, *Pattern Analysis and Applications* **2** (1999), 59–72.

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

[16] N. Boujemaa, On Competitive Unsupervised Clustering. In the International Conference on Pattern Recognition (ICPR'00), vol. 1, 1631–1634.

[17] C. Carpineto and G. Romano, A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval, *Machine Learning* **24**(2) (1996), 95–122.

[18] Y. Cheng and G. Church, Biclustering of Expression Data. In *Proceedings of the 8^{th} International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, 2000, 93–103.

[19] S. Chu and J. Roddick, A Clustering Algorithm Using Tabu Search Approach with Simulated Annealing for Vector Quantization, *Chinese Journal of Electronics* **12**(3) (2003), 349–353.

[20] F. Chung, *Spectral Graph Theory*, Society Press, 1997.

[21] S. Cohen and L. de Castro, Data Clustering with Particle Swarms. In *Proceedings of the IEEE Congress on Evolutionary Computation*. Vancouver, BC, Canada, 2006, 1792–1798.

[22] G. Coleman and H. Andrews, Image Segmentation by Clustering. In *Proceedings of IEEE*, vol. 67, 1979, 773–785.

[23] D. Comaniciu and P. Meer, Mean Shift: A Robust Approach Toward Feature Space Analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(5) (2002), 603–619.

[24] H. Dai and W. Ma, A Novelty Bayesian Method for Unsupervised Learning of Finite Mixture Models. In *Proceedings of the 3^{rd} International Conference on Machine Learning and Cybernetics*, Shanghai, China, 2004, 3574–3578.

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

[26] D. Davies and D. Bouldin, A Cluster Separation Measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **1**(2) (1979).

[27] 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.

[28] F. Divina and J. Aguilar-Ruiz, Biclustering of Expression Data with Evolutionary Computation, *IEEE Transactions on Knowledge and Data Engineering* **18**(5) (2006), 590–602.

[29] J.C. Dunn, Well Separated Clusters and Optimal Fuzzy Partitions, *Journal of Cybernetics* **4** (1974), 95–104.

[30] B. Everitt, *Cluster Analysis*, Heinemann Books, London, 1974.

[31] B. Everitt, S. Landau and M. Leese, *Cluster Analysis*, A Hodder Arnold Publication, 4th edition, 2001.

[32] A. Fielding, *Cluster and Classification Techniques for the Biosciences*, Cambridge Press, 2007.

[33] M. Figueiredo and A. Jain, Unsupervised Learning of Finite Mixture Models, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(3) (2002), 381–396.

[34] E. Forgy, Cluster Analysis of Multivariate Data: Efficiency versus Interpretability of Classification, *Biometrics* **21** (1965), 768–769.

[35] H. Frigui and R. Krishnapuram, Clustering by Competitive Agglomeration, *Pattern Recognition Letters* **30**(7) (1997), 1109–1119.

[36] H. Frigui and R. Krishnapuram, A Robust Competitive Clustering Algorithm with Applications in Computer Vision, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **21**(5) (1999), 450–465.

[37] P. Frnti, J. Kivijrvi and O. Nevalainen, Tabu Search Algorithm for Codebook Generation in Vector Quantization, *Pattern Recognition* **31**(8) (1998), 1139–1148.

[38] I. Gath and A. Geva, Unsupervised Optimal Fuzzy Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**(7) (1989), 773–781.

[39] F. Glover, Tabu Search – Part I, *ORSA Journal on Computing* **1**(3) (1989), 190–206.

[40] F. Glover, Tabu Search – Part II, *ORSA Journal on Computing* **2**(1) (1990), 4–32.

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

[42] M. Halkidi, Y. Batistakis and M. Vazirgiannis, On Clustering Validation Techniques, *Intelligent Information Systems Journal* **17**(2–3) (2001), 107–145, Kluwer Pulishers.

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

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

[45] J. Hartigan. Direct Clustering of Data Matrix, *Journal of the American Statistical Association* **67**(337) (1972), 123–129.

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

[47] 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*), 2002, 600–607.

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

[49] K. Huang, A Synergistic Automatic Clustering Technique (Syneract) for Multispectral Image Analysis, *Photogrammetric Engineering and Remote Sensing* **1**(1) (2002), 33–40.

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

[51] A. Jain, R. Duin and J. Mao, Statistical Pattern Recognition: A Review, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(1) (2000), 4–37.

[52] A. Jain, M. Murty and P. Flynn, Data Clustering: A Review, *ACM Computing Surveys* **31**(3) (1999), 264–323.

[53] D. Judd, P. Mckinley and A. Jain, Large-scale Parallel Data Clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**(8) (1998), 871–876.

[54] L. Kaufman and P. Rousseuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & sons, Inc., New York, 1990.

[55] 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* **90**(431) (1995), 928–934.

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

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

[58] R. Klein and R. Dubes, Experiments in Projection and Clustering by Simulated Annealing, *Pattern Recognition* **22** (1989), 213–220.

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

[60] Krishnapuram and Keller, A Possibilistic Approach to Clustering, *IEEE Transactions on Fuzzy Systems* **1**(2) (1993), 98–110.

[61] Krishnapuram and Keller, The Possibilistic C-Means algorithm: Insights and Recommendations, *IEEE Transactions on Fuzzy Systems* **4**(3) (1996), 385–393.

[62] N. Labroche, N. Monmarche and G. Venturini, Visual Clustering based on the Chemical Recognition System on Ants, In *Proceedings of the European Conference on Artificial Intelligence*, Lyons, France, 2002.

[63] D. Langan, J. Modestino and J. Zhang, Cluster Validation for Unsupervised Stochastic Model-Based Image Segmentation, *IEEE Transactions on Image Processing* **7**(2) (1998), 180–195.

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

[65] Y. Leung, J. Zhang and Z. Xu, Clustering by Space-Space Filtering, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(12) (2000), 1396–1410.

[66] 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.

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

[68] S. Lu and K. Fu, A Sentence-to-Sentence Clustering Procedure for Pattern Analysis, *IEEE Transaction on Systems, Man and Cybernetics* **8** (1978), 381–389.

[69] E. Lumer and B. Faieta, Diversity and Adaptation in Populations of Clustering Ants. In *Proceedings of the Third International Conference on Simulation and Adaptive Behavior*, 1994, 501–508.

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

[71] U. Maulik and S. Bandyopadhyay, Genetic Algorithm-Based Clustering Technique, *Pattern Recognition* **33** (2000), 1455–1465.

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

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

[74] N. Merhav, The Estimation of the Model Order in Exponential Families, *IEEE Transactions on Information Theory* **35** (Sep. 1989).

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

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

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

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

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

[80] 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.

[81] M. Omran, A. Engelbrecht and A. Salman, Particle Swarm Optimization Method for Image Clustering, *International Journal of Pattern Recognition and Artificial Intelligence* **19**(3) (May 2005), 297–322.

[82] M. Omran and A. Engelbrecht, Self-Adaptive Differential Evolution Methods for Unsupervised Image Classification. In the *Proceedings of the IEEE International Conferences on Cybernetics and Intelligent Systems* (*CIS 2006*), Bangkok, Thailand, IEEE Press, June 2006, 1–6.

[83] M. Omran, A. Salman and A. Engelbrecht, Self-adaptive differential evolution, In *the proceedings of The 2005 International Conference on Computational Intelligence and Security,* December, Xi'an, China, Springer's Lecture Notes in Artificial Intelligence, p. 3801, 2005, 192–199.

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

[85] N. Pal and J. Biswas, Cluster Validation using Graph Theoretic Concepts, *Pattern Recognition* **30**(6) (1997).

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

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

[88] 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*, Morgan Kaufmann, San Francisco, CA, 2000, 727–734.

[89] 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*, 1979, 10–22.

[90] R. Rendner and H. Walker, Mixture Densities, Maximum Likelihood and the EM Algorithm, *SIAM Review* **26**(2) (1984).

[91] J. Rissanen, Modeling by Shortest Data Description, *Automatica* **14** (1978), 465–471.

[92] S. Roberts, D. Husmeier, L. Rezek and W. Penny, Bayesian Approaches to Gaussian Mixture Modeling, *IEEE Transactions in Pattern Recognition and Machine Intelligence* **20**(11) (1998), 1133–1142.

[93] 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, 2000, 1656–1659.

[94] P. Scheunders, A Comparison of Clustering Algorithms Applied to Color Image Quantization, *Pattern Recognition Letters* **18**(11–13) (1997), 1379–1384.

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

[96] 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.

[97] 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).

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

[99] J. Tou, DYNOC – A Dynamic Optimal Cluster-seeking Technique. *International Journal of Computer and Information Sciences* **8**(6) (1979), 541–547.

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

[101] D. Van der Merwe and A. Engelbrecht, Data Clustering using Particle Swarm Optimization, In *Proceedings of the IEEE Congress on Evolutionary Computation*, Canberra, Australia, 2003, 215–222.

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

[103] C. Veenman, M. Reinders and E. Backer, A Maximum Variance Cluster Algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(9) (2002), 1273–1280.

[104] C. Veenman, M. Reinders and E. Backer, A Cellular Coevolutionary Algorithm for Image Segmentation, *IEEE Transactions on Image Processing* **12**(3) (2003), 304–316.

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

[106] C. Wallace and D. Boulton, An Information Measure for Classification, *The Computer Journal* **11** (1968), 185–194.

[107] 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, 1994, 37–44.

[108] B. Wu and Z. Shi, A Clustering Algorithm based on Swarm Intelligence. In *Proceedings of the International Conference on Info-tech and Info-net*, 2001, 58–66.

[109] Z. Xiang, Color Image Quantization by Minimizing the Maximum Inter-cluster Distance, *ACM Transactions on Graphics* **16**(3) (1997) 260–276.

[110] X. Xiao, E. Dow, R. Eberhart, Z. Ben Miled and R. Oppelt, Gene Clustering using Self-Organizing Maps and Particle Swarm Optimization. In *Proceedings of the 2^{nd} IEEE Congress on Evolutionary Computation*, Canbella, Australia, 2003, 215–220.

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

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

[113] Z. Zivkovic and F. van der Heijden, Recursive Unsupervised Learning of Finite Mixture Models, *IEEE Transactions of Pattern Analysis and Machine Intelligence* **26**(5) (2004), 651–656.